DESIGN METHODOLOGY FOR 3D-STACKED IMAGING SYSTEMS WITH INTEGRATED DEEP LEARNING

A Dissertation
Presented to
The Academic Faculty

by

Mohammad Faisal Amir

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
August 2018

DESIGN METHODOLOGY FOR 3D-STACKED IMAGING SYSTEMS WITH INTEGRATED DEEP LEARNING

Approved by:

Dr. Saibal Mukhopadhyay, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Tushar Krishna
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Sudhakar Yalamanchili
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Paul Kohl
School of Chemical & Biomolecular Engineering
Georgia Institute of Technology

Dr. Asif Islam Khan
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: April 25, 2018

ACKNOWLEDGEMENTS

I would like to thank my advisor, Professor Saibal Mukhopadhyay, for his continuous support and encouragement throughout the course of my graduate studies. His unique insights were invaluable towards helping me develop a better understanding of how to approach a multitude of research problems. Without his unwavering confidence in my abilities (even at times when I did not have faith in myself), this work would not have been completed.

I would also like to thank my committee members, Professor Asif Islam Khan, Professor Paul Kohl, Professor Tushar Krishna, and Professor Sudhakar Yalamanchili for their valuable time and suggestions towards improving the quality of this thesis.

Thanks are due to my amazing colleagues at GREEN lab for all the help, discussions, and collaborations I have had over the last few years. I am grateful to have worked with and known Dr. Subho Chatterjee, Dr. Denny Lie, Dr. Amit Trivedi, Dr. Boris Alexandrov, Dr. Wen Yueh, Dr. Sergio Carlo, Dr. Jaeha Kung, Dr. Duckhwan Kim, and Dr. Monodeep Kar. I am also thankful to the current students Taesik Na, Arvind Singh, Yun Long, Burhan Mudassar, Edward Lee, Venkata Chaitanya Krishna Chekuri, Nikhil Chawla, Nihar Dasari, Priyabrata Saha, and Minah Lee. In particular, special thanks are due to Dr. Khondker Zakir Ahmed for his mentorship in both academic and personal affairs, and to Dr. Jong Hwan Ko for his support all throughout the course of our four-year-long collaborative research.

I would like to acknowledge my family for supporting me in all aspects of my life. Thanks to my parents, Mohammad Amirul Islam and Momtaz Islam, and my brother,

Mohammad Ashraful Islam; without your guidance and encouragement during my formative years, I would not be where I am today. Finally, thanks are due to my muse and partner-in-crime, Syeda Faria Tus Sadia, who put her entire life on hold while I set about to pursue my dreams. Her cheerful personality and unwavering optimism made sure the frustrating periods of uncertainty and depression were few and far between over the last few years. There is no one else I would rather undertake this journey with, and this degree is as much mine as it is yours.

Above all, I am grateful to the Almighty for blessing me with the patience, intellect, and resolve necessary for completing this difficult challenge.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The Internet of Things (IoT) revolution has brought along with it billions of always on, always connected devices and sensors, associated with which are huge amounts of data that must be transmitted to an off-chip host for classification. However, sending these large volumes of unprocessed data incurs large latency and energy penalties which impairs the energy efficiency of resource constrained IoT systems. Moving computations to the sensor offers the potential to improve performance and energy efficiency of the end application.

The objective of the presented research is to explore sensor integrated computing which allows the deployment of smart sensors capable of performing computations in-field. Initially, we introduce the design of a 3D-stacked image sensor with integrated deep learning, which uses the advantages of 3D integration to increase sensor fill factor, simplify routing, increase parallelism, and enhance memory capacity. Through an exploration of the design space we investigate how the system architecture and resource constraints can dictate system metrics such as the optimum energy efficiency configuration and accuracy-throughput tradeoffs. Next, we examine technology based solutions to further enhance system performance through the use of 3D stacked digital sensors with in-pixel ADCs, and explore how emerging device based processing-in-memory neural accelerators can offer superior energy efficiency. Furthermore, the various circuit issues involved with the design of these sensor based systems are investigated through the discussion of post-silicon results from an image sensor SOC with integrated energy harvesting. The dissertation concludes with a discussion on how energy harvesting sensors can be used to achieve energy neutral self-powered systems capable of operating solely with harvested energy.

CHAPTER 1. INTRODUCTION

Connected pervasive devices with sensing, processing and communication capabilities, often referred to as the Internet of Things (IoT), are emerging as a key driver of the future growth of electronics. From smart home automation hubs to connected appliances to wearable technology, IoT is set to permeate and connect every aspect of our lives. In an IoT environment, a sensor (edge) captures and sends data to a distant processing engine (host) where the actual processing and computation take place. The problem with such an approach is that no in-field decision making takes place, and sending unprocessed (large volume) data to the host typically incurs large latency and energy penalties, especially for low-bandwidth channel as in remote sensing via wireless networks. Moving a part, or the entirety, of these computations on to the sensor side will offer energy efficiency advantages while simultaneously improving throughput.

3D integration has shown significant benefit for design of image sensors. 3D stacking of photodiode and read-out circuits (e.g. ADC) in separate layers facilitates very high-speed imaging by enabling parallel data transfer between pixel array and ADC. Moreover, eliminating peripheral circuits from the pixel-array layer also significantly increases the fill factor of imagers. On the processing side, recent advances in deep neural networks have demonstrated significant success in solving complex computer classification problems such as image classification, and 3D integration also shows promise for the implementation of these neural networks on chip. 3D stacking of memory with a specialized logic layer for neural computation results in energy-efficient deep neural network (DNN) engines by enabling high bandwidth and concurrent access between

compute and data (synaptic weights and neuron states). The highly concurrent memory access successfully leverages the highly parallel nature of neural computations.

The primary focus of this thesis is to explore the interaction between architecture, technology, and circuits, and delve into how these parameters impact the system level design and performance of sensor based systems.

1.1 Thesis Objective and Organization

The objective of the presented research is to explore sensor integrated computing which allows the deployment of smart sensors capable of performing computations in-field. Initially, we introduce the design of a 3D-stacked image sensor with integrated deep learning, which uses the advantages of 3D integration to increase sensor fill factor, simplify routing, increase parallelism, and enhance memory capacity. Through an exploration of the design space we investigate how the system architecture and resource constraints can dictate system metrics such as the optimum energy efficiency configuration and accuracy-throughput tradeoffs. Next, we examine technology based solutions to further enhance system performance through the use of 3D stacked digital sensors with in-pixel ADCs, and explore how emerging device based processing-in-memory neural accelerators can offer superior energy efficiency. Furthermore, the various circuit issues involved with the design of these sensor based systems are investigated through the discussion of post-silicon results from an image sensor SOC with integrated energy harvesting. The dissertation concludes with a discussion on how energy harvesting sensors can be used to achieve energy neutral self-powered systems capable of operating solely with harvested energy. The dissertation is organized as follows.

CHAPTER 2 establishes background and goes over the previously published literature relevant to the presented research.

CHAPTER 3 presents the basic Neurosensor architecture and lays out the design details of the various component blocks. Detailed power and performance analysis is performed for two configurations of the system under limited (sensor-host) bandwidth scenarios, and DNN partitioning for optimum energy efficiency is explored. In addition, a noise model for the sensor is developed, and the associated trade-offs between system throughput and neural network classification accuracy is investigated.

CHAPTER 4 extends the work presented in the previous chapter, and explores the design of a massively parallel, high throughput digital image sensor. To take advantage of the high throughput enabled by digital pixels, an emerging device based processing-in-memory (PIM) architecture is investigated as a possible option for the neural accelerator tier. The impact of in-memory computation as well as digital sensors are evaluated through power and performance analysis for various configurations of the system.

CHAPTER 5 introduces the design of a 2D image sensor SOC with an image sensor array that can be configured to operate in either imaging or harvesting mode. The sensor captures, converts and compresses images under imaging mode, and under harvesting mode, turns effectively into a solar cell from which energy can be harvested. Post silicon results from the chip are presented, along with a discussion about how the various shortcomings can be overcome. Finally, measurement results from a revised version of the SOC, based on findings from the previous chip, are presented.

CHAPTER 6 offers concluding remarks, summarizes the dissertation, and includes a brief discussion on future research direction.

CHAPTER 2. BACKGROUND & LITERATURE SURVEY

2.1 3D Integrated Image Sensors

3D integration offers numerous advantages which include increased parallelism, wide bandwidth interconnects, decreased system footprint, and routing overhead reduction (leading to latency and power optimization) [1-4]. While all these advantages carry over to image sensors, the principal advantage of 3D integration in the context of image sensors is that it can enable high fill factor imagers [5]. This can be accomplished by having the top tier composed entirely of pixels (or photodiodes) and pushing the analog to digital converters and/or readout circuits to the bottom tier(s) [6, 7]. Therefore for the same footprint as a 2D configuration, increased resolution can be achieved. Another added advantage of 3D stacking is the opportunity for heterogeneous integration [8-10], which allows the fabrication of the imager and A/D conversion layers in different process nodes, thus combining optimal photosensitivity (for imager tier) with scaling advantages (for the other tiers). In this section we are going to briefly go over the numerous works investigating various design methodologies and configurations for 3D image sensor design.

The authors of [11-13] presented a block parallel image sensing and processing architecture which consisted of a CMOS image sensor (CIS) layer, Correlated Double Sampling (CDS) layer, and ADC layer, connected through back-side TSVs. Each image frame was 320×240 pixels divided into 20×15 image processing blocks, with all blocks operating in parallel. Each block contained 255 pixels, one CDS circuit and one ADC circuit connected through TSV. This system also used heterogenous integration, with the ADC layer being fabricated in 90nm technology and the other layers at 180nm.

Zhang et al. [14] demonstrated one of the first examples of 3D image sensors with integrated feature extraction. The imager contained 64×96 pixels, fabricated in 180nm

FDSOI process, and consisted of three tiers connected through inter-tier vias – with the top tier being photodiodes, the second tier containing pixel reset transistor, in-pixel buffer and row and column scanners, and the bottom tier containing computation circuits including analog memory and resistor network. Although this system could perform contour extraction or temporal differencing, a shortcoming of this system was that feature extraction required off-chip subtraction.

Sukegawa et al. [15] demonstrated one of the first mainstream commercial 3D image sensors (Sony Exmor RS series). The imager had an effective resolution of 8MP, and contained two tiers – the top tier contained back-illuminated CIS pixels and ADC comparators, fabricated in 90nm process, and the bottom tier contained rest of the ADC as well as row and column circuitry. The tiers were connected through TSVs, and the number of TSVs approximately equaled the number of row and column signals, which are of the order of thousands. This chip also contained integrated image processing functionality, which enabled HDR movie recording and increased sensitivity.

The authors of [16, 17] investigated the design of a CMOS image sensor with 3D stacked image compression unit, with the image sensor being designed in 180nm and heterogeneous integration (with various technologies) being explored for the other tiers. The system contained five layers (CIS, column circuits, ADC, image buffer, and compression unit), and investigated multi-segment image compression to increase parallelism and throughput. The paper also performed thermal analysis on the system, and studied the effects of temperature induced noise on image quality.

Haruta et al. [18] introduced the first commercial 3-layer stacked CMOS image sensor. In addition to the standard approach of separating the pixels and circuits onto two different stacks, this sensor also contained a third layer consisting of 1 Gb DRAM. Compared to a sensor without DRAM whose speed is limited by the processing and

interface speed, stacking DRAM on the sensor itself allows large-size frame data to be read out from the pixels and stored on the DRAM at high speeds, and then sent to the main processing unit at usual speeds. The sensor could read out 19MP at 120fps and output at 30fps. In addition, the DRAM provided additional capabilities to the sensor, such as super-slow motion (960 fps) and electronic image stabilization.

2.1.1 3D Integrated Sensors with In-Pixel ADC

A class of 3D image sensors that has been gaining traction in recent years consist of "digital" pixels which employ 3D integration to place dedicated ADCs for each pixel in the bottom layer [19]. This enables massive parallelism by enabling all the pixels to be read out simultaneously. One such work [20] implemented a 128×96 pixel array which used Au electrodes to connect the pixel and ADC layers. The sensor used a simple pulse-based architecture for A/D conversion which was compact in area and yet provided 96dB dynamic range.

Sakakibara et al. [21] demonstrated a 1.46MP sensor array with pixel-parallel ADC with global shutter function which eliminated the image distortion caused due to the conventional row-by-row readout mechanism. Instead of TSVs, the sensor used Cu-Cu connections to bond the two dies which provided more freedom in design, allowed for a more compact size, and increased performance. A new readout circuit was also developed to support the massively parallel data transfer required to simultaneously read and write all the pixel signals at high speed. In addition, to minimize the power draw due to the inclusion of such a large number of ADCs, the sensor used a new comparator design in the ADCs which operates on subthreshold current.

2.2 3D Integrated Neural Networks

Neural networks have recently seen a new surge in interest due to their widespread

applications in machine learning. Neural accelerators are well suited to 3D stacking because they are inherently 3D structures characterized by the movement of high bandwidth data from layer to layer [22-25]. Mapping such a 3D structure onto 2D circuits results in routing difficulties and/or long latencies. In addition, computations in neural networks lend themselves to a large degree of parallelism, which allows the system to run at a low clock frequency, thus reducing thermal concerns while still achieving high throughput. There have recently been quite a few works on 3D neural accelerators, and 3D integration shows great promise for the design of neuromorphic hardware.

Belhadj et al. [26] presented one of the first examples of a 3D stacked neural accelerator. As a proof of concept, a 2 layer spiking neural network was designed in 130nm, using microbumps to bond the two layers. This neural accelerator was meant to be used as a pre-processor, placing it between CMOS (image) sensor and processor, which overcame the memory limitations associated with fetching data from memory when neural accelerators are used as co-processors. The accelerator was designed to process a 128×96 pixel image, with the first layer performing feature extraction (using 48 neurons) and the second layer performing classification (using 50 neurons). For comparison, the same neural network was also implemented in 2D, and it was found that the 2D circuit consumed 27% more power than the 3D version, and its cycle time was 36% higher, primarily due to the much larger routing network between the two layers. Overall, the 3D circuit required only 0.48× of the energy required to process a single input image compared to the 2D version.

Kim et al. [27] introduced Neurocube, a programmable digital neuromorphic architecture based on Micron's Hybrid Memory Cube [28], which was essentially a variable number (four in this case) of stacked DRAM dies, followed by a bottom logic die.

The architecture was partitioned into 16 units called vaults, which lent itself well to parallelism. One of the key points for this work was that it offered a programmable architecture through the use of Programmable Neurosequence Generators (PNG), thus allowing it to maintain flexibility while achieving better energy efficiency compared to GPU. The paper also explored various architectural details including the design and programming of the PNG, as well as data movement through the network. As an example test case, scene labeling [29] was implemented on the Neurocube to study system hardware requirements, throughput and power consumption. In addition, changes in system performance with different logic die process nodes was investigated, and thermal simulation performed to ensure that the system stayed within the thermal budget.

The authors of TETRIS [30] also demonstrated a 3D neuromorphic architecture based on HMC. Their analysis showed that the high throughput and low energy associated with 3D memory allows the rebalancing of NN accelerator design, using more area for logic and less area for SRAM buffer. In addition, portions of the NN computations were moved onto the DRAM dies to decrease bandwidth pressure and increase performance and energy efficiency. A partitioning scheme was also developed which allowed the parallelization of NN computations over multiple vaults. Compared to a conventional 2D NN accelerator, the design achieved 4× performance improvement with 1.5× energy saving.

Neurostream [31] proposed a scalable and energy-efficient processor in memory system for the execution of deep convolutional networks based on a network of connected Smart Memory Cubes (SMC), which are essentially modular extensions to the standard HMC. In addition, the HMC logic dies were also augmented with a many-core PIM

platform called the Neurocluster, which increased the logic die area by only 8% while achieving an average performance of 240 GFLOPS for complete classification within a 2.5W power budget. The work also demonstrated the possibility of scaling the performance to 955 GFLOPS by interconnecting a network of four SMCs.

2.3 Emerging Device Based Neural Accelerators

Since memory access is one of the principal components of latency and energy consumption in neural accelerators, integrating computation and storage within a memory device offers intriguing opportunities to enable processing in memory (PIM) computing and eradicate the separation between computation and data, thus leading to throughput and energy efficiency advantages. There have recently been considerable advances in implementing neural accelerators using emerging devices such as ReRAM [32]. The principal methodology behind these architectures is to use a crossbar array to perform vector-matrix multiplication using mixed signal computations.

ISAAC [33] demonstrated the promise of using ReRAM crossbar arrays to simultaneously store data and perform neural computations in memory. The work designed a pipelined architecture for neural acceleration, with dedicated crossbars for each neural network layer; eDRAM buffers were used to combine data between pipeline stages. New data encoding techniques were also defined to reduce A/D conversion overhead. System throughput and power analysis were carried out to identify the optimum balance of ReRAM storage/compute, ADCs and on-chip eDRAM storage. Compared to the state-of-the art, ISAAC achieved improvements of 14.8×, 5.5×, and 7.5× in throughput, energy, and computation density respectively.

Chi et al. presented PRIME [34], a PIM architecture to accelerate NN computations using ReRAM memory. The architecture allowed a portion of the ReRAM crossbar array to be configured as either as neural computation accelerator, or as regular memory, thus enabling the PIM architecture. Circuit and microarchitecture innovations enabled the morphological architecture with minimum area overhead. Compared to current state of the art neural processing units, the work reported 2360× performance improvement and reduced energy consumption by 895× due to the PIM architecture and efficient NN computation using ReRAM.

Pipelayer [35] improved upon ISAAC [33] by designing a pipelined architecture which supported both online training and testing. Pipelayer also adopted a different pipeline architecture from ISAAC so that data could continuously flow into the accelerator in consecutive cycles, which is essential to support pipelined training. It also minimized the ADC and DAC overhead with spike-based integration and fire circuit based design for data input. Compared to GPU platform, Pipelayer achieved 42.45× speedup and 7.17× energy saving.

2.4 Energy Harvesting Image Sensors

Since the primary component in both image sensors and solar cells are photodiodes, an interesting opportunity arises to harvest energy from image sensors, and some interesting concepts have emerged in recent years [36]. Fish et al. [37] introduced an energy harvesting pixel with an extra power generation photodiode which harvested energy. A problem with this approach was that the power generation photodiode occupied significant area which reduced the amount of area available for the image sensing photodiode. The

authors of [38] used two separate photodiodes for power generation and image capture to design a time to first spike image sensor architecture. When the pixels fired, they were reconfigured so that the image capture diode was connected in parallel with the power generation photodiode, thus increasing the amount of harvested energy. However the power generation photodiode still occupied a large percentage of pixel area, and the time to first spike architecture required additional circuits inside the pixel.

Law et al. [39] presented a configurable energy harvesting pixel design that used a p-diffusion/n-well photodiode and seven transistors to configure the pixel in either energy harvesting or image sensing mode. To configure the photodiode as an energy harvester, its cathode was connected to ground via an in-pixel transistor. A similar concept was employed in [40], however, the pixel used two photodiodes and four transistors.

A potential problem with the architecture in [39, 40] was that connecting the cathode to ground via a transistor activated the parasitic n-well/p-sub photodiode which diverted photo-generated charges away from the p-diffusion/n-well photodiode, thus reducing harvesting efficiency. Wang et al. [41] addressed this problem by employing a DC-DC converter with a flying inductor. A flying inductor kept the n-well floating, which turned off the parasitic photodiode. The pixel included a 1 bit memory which allowed each pixel to be individually configured as an imager or a harvester. A downside of this design was that each pixel contained 12 transistors, thus reducing the fill factor.

CHAPTER 3. NEUROSENSOR: 3D IMAGE SENSOR WITH INTEGRATED DEEP LEARNING

3.1 System Overview

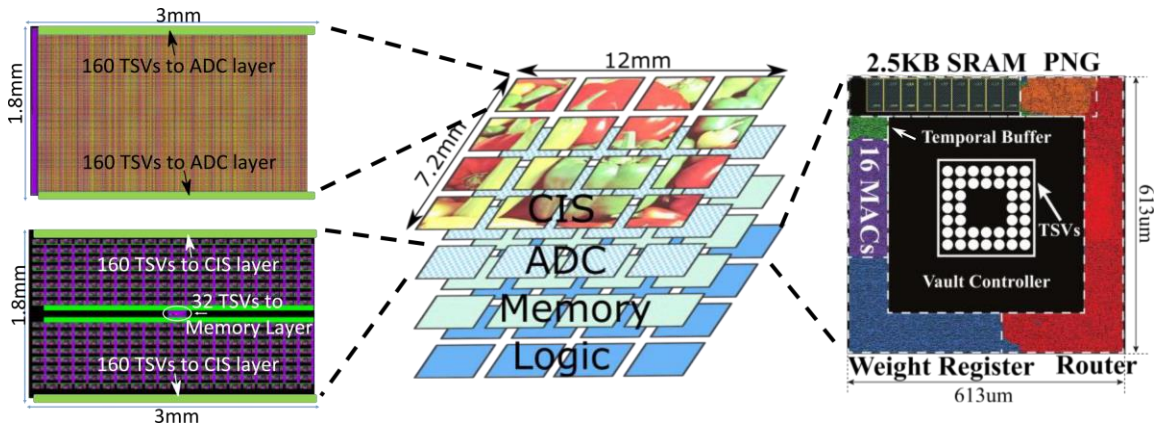


Figure 1 Schematic of 3D stacked image sensor with integrated neural logic

Figure 1 shows a schematic overview of our 3D integrated system. The top layer consists of an HD (1280×768) image sensor, the second layer is composed of an array of 8-bit single-slope ADCs which convert the image obtained from the image sensor layer. The next layer(s) contains memory to store the image as well as synaptic weights for neural logic (there can be more than one memory layer for large memory requirements, as we will see in the subsequent sections). Finally, the bottom layer contains the neural logic. The total system footprint for a single layer is 12mm×7.2mm, determined by the image sensor tier, and the tiers connect to one another through TSVs. In order to increase parallelism, the whole system is arranged in a 4×4 grid, with all the 16 segments working in parallel. We also employ heterogeneous integration, designing the image sensor and ADC in 130nm and the logic layer in 15nm. This allows us to leverage the superior optical and analog

performance offered by the larger process nodes, while keeping the performance, density and energy efficiency offered by digital circuits in scaled technologies [42]. The data flow consists of the following stages

1. Sensing – The photodiode layer captures the image and passes the analog signals to the ADCs in the layer below; the signal in each column line is propagated through a dedicated TSV. In the second layer, there is one 8-bit ADC for each column line in the pixel array, resulting in a total of 5120 ADCs (320 ADCs per segment, 16 parallel segments). For each segment, these ADCs perform digital conversion on the analog signals from the photodiode layer, and multiplex the converted digital data through a 32-bit bus into the memory tier below.

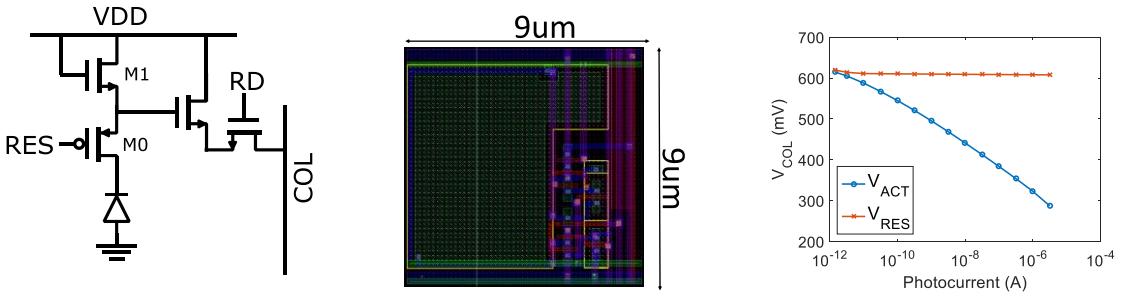
2. Neural Computation – The converted images, stored in memory, are passed on to the logic layer below where they undergo neural computation. There are multiple options that can be implemented for the neural computation layer with different degrees of complexity, memory requirements, and throughput. There may be some configurations that only perform a few levels of feature extraction on the image, leaving the rest of the computation to the off-chip host, or the entire neural network may be implemented on chip, thus performing full neural classification and transmitting only the classification data to the host.

3. Transmission – Once the neural layer performs processing on the image, it can be transmitted through a wireless transceiver to an off-chip host. This completes the data flow for a single frame, and work on the next frame can start after the previous frame has been

successfully transmitted. It should be noted that we do not model the transmitter since we consider it to be an off-chip module.

In the subsequent sections, we will go over details of the sensor framework and the neural network platform.

3.1.1 CMOS Image Sensor Tier



(a) **(b)** **(c)**

Figure 2 (a) Circuit schematic of logarithmic pixel with CDS (b) Pixel layout (c) Column voltage response to changing photocurrent at 25°C

Figure 2 (a) and (b) show the circuit schematic and layout of the logarithmic pixel [43] used in the array, and the layout for a single segment (320×192 pixels) of the CIS layer, designed in 130nm, can be seen in Figure 1. The segment of 320×192 pixels is stamped in a 4×4 grid to create the imaging array with a resolution of 1280×768 pixels. In order to reduce time-invariant spatial noise due to transistor variation, commonly referred to as fixed pattern noise (FPN) [44], the pixel uses correlated double sampling (CDS). For CDS two samples are taken for every cycle, one at the reset phase with RES signal high (which turns off transistor M0 and simulates dark condition), and another at the active phase with RES signal low (which turns on transistor M0 and equates to an illuminated sample). The voltages at the reset and active phases can be approximated by

$$V_{RES} = VDD - V_{th,M1} - nV_T \ln\left(\frac{I_{leak,M0}}{I_0}\right) \quad (1)$$

$$V_{ACT} = VDD - V_{th,M1} - nV_T \ln\left(\frac{I_{ph}}{I_0}\right) \quad (2)$$

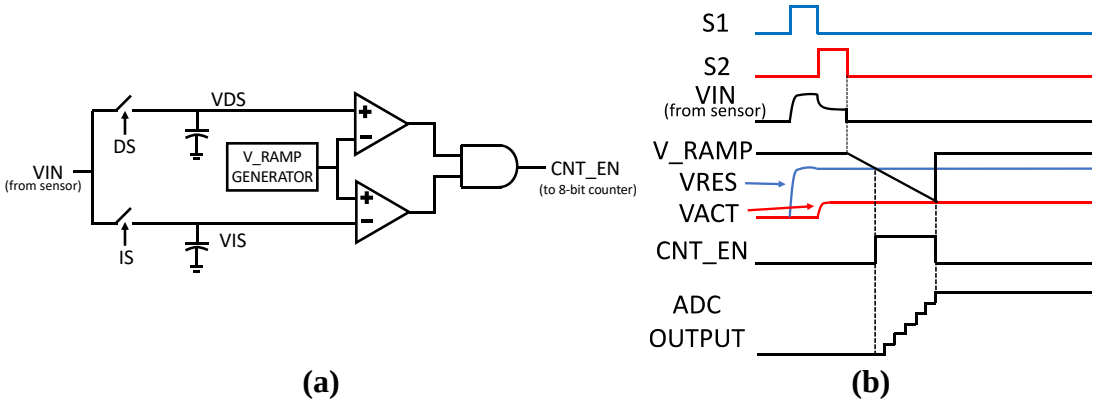
where V_{RES} and V_{ACT} are the output voltages when RES is high and low respectively, I_{leak} is the leakage current through M0, I_0 is the subthreshold current of M1, V_T is the thermal voltage, Vth is the transistor threshold voltage, n is the process slope factor and I_{ph} is the illumination induced photocurrent. The difference between (1) and (2) equates to

$$\Delta V = V_{ACT} - V_{RES} = nV_T \ln\left(\frac{I_{ph}}{I_{leak,M0}}\right) \quad (3)$$

From (3), it can be seen that the above scheme eliminates variation due to transistor M1, which is responsible for the majority of the fixed pattern noise (FPN). Thus our readout scheme actually samples twice and it is the difference of the two samples that undergoes A/D conversion. Figure 2(c) shows how the column voltages, V_{ACT} and V_{RES}, change with photocurrent.

3.1.2 Analog to Digital Converter (ADC) Tier

The ADC layer, also designed in 130nm, is situated directly below the image sensor layer and serves to digitally convert the image signal from the sensor layer above it. Each column line has a dedicated 8-bit single slope ramp ADC [45], equating to 320 ADCs per segment and a total of 5120 ADCs, all of which operate in parallel during conversion. The outputs of the ADCs are multiplexed and serialized on to the memory layer placed below.



(a) **(b)**

Figure 3 (a) Block diagram representation of ADC (b) Waveforms for ADC readout operation

Figure 3(a) shows a block diagram representation of the 8-bit single-slope ADC, similar to that used in [46], and Figure 3(b) shows a waveform of the signal readout. For every A/D conversion cycle, the signals DS and IS sample the voltages V_{RES} and V_{ACT} and store them onto their respective sampling capacitors. Once the sampling is completed, a ramp waveform is generated. The ADC comparators are configured in such a way that the 8-bit counter is turned on only when the ramp voltage, V_{RAMP}, lies between V_{RES} and V_{ACT}. Thus only the voltage difference between V_{ACT} and V_{RES} undergoes A/D conversion. In order to ensure proper matching with the CMOS pixels, the ADCs are also designed in 130nm.

3.1.3 Neural Logic Tier

Figure 4 shows the major components of the neural architecture in the logic tier (based on [27]), synthesized using 15nm FinFETs [47], and the layout can be seen in Figure 1. The system consists of a global controller, processing elements (PE), routers for a 2D mesh network on chip (NoC), and programmable neurosequence generator (PNG) for

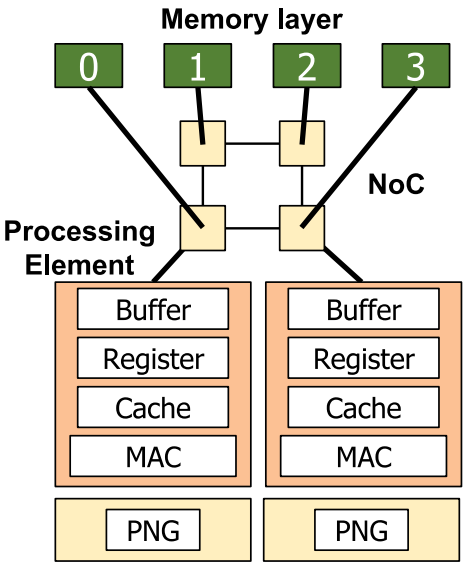


Figure 4 Simplified architecture of the neural logic tier

DRAM. The PEs communicate with the memory tier through high-speed TSVs, and all PEs are connected by a 2D mesh network.

The processing element (PE) is the main computing unit and consists of 16 multiply accumulator (MAC) units, an SRAM cache memory, a temporal buffer, and a register module for storing synaptic weights. The state of the input neurons and associated connectivity weights are encapsulated in a packet and moved to the PEs by the Programmable Neurosequence Generators (PNG). If packets arrive out of order, they are buffered in SRAM cache until proper order is restored. When all corresponding inputs arrive, they are moved to the temporal buffer and a MAC operation is triggered.

A 2D mesh network connects all the PEs and enables communication with the other segments to enable a parallel core architecture. It also allows the logic tier to communicate with the memory layer above it and transfer image data as well as synaptic weights. Each PE is connected to a single router, which uses deterministic X-Y routing [48], and each

router has 6 input and 6 output channels (4 for neighboring routers and 2 for PE and memory).

Each segment in the logic layer has an associated PNG that dictates the data movements required for neural computations. For a given neuron in a layer, the PNG generates a sequence of addresses for the neurons in the previous layer that are connected to it, as well as the corresponding synaptic weights between them. This operation is then repeated for each neuron in the layer. The data corresponding to each neuron is encapsulated in a packet and transferred to the corresponding PE through the router in the NoC. The PNG can be programmed externally by the host to enable a wide range of neural architectures. More details about the neural logic layer can be found in [27].

3.2 CNN Architectures and Organization

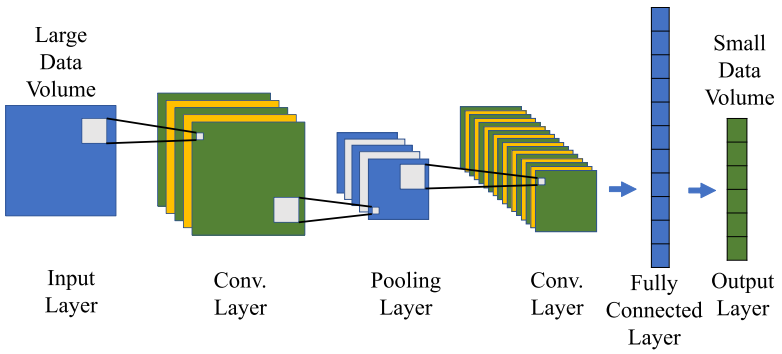


Figure 5 Conventional CNN architecture

Due to their efficacy in image classification tasks [49], we concern ourselves primarily with convolutional neural networks (CNN) [50] for this work. Figure 5 shows a conventional CNN architecture, which generally contains a number of successive convolution and max-pooling layers followed by fully connected layers. The convolutional layers contain K kernels with size $m \times n$ that filter the data by performing 2D convolution, and the filtered response is then subsampled into the pooling layer. These convolution and

pooling layers perform feature extraction and are laid alternately to create a deep network architecture. Finally, the fully connected layers output the probability of each class.

The first CNN that we study is AlexNet [51], which is arguably the neural network that brought CNNs to the forefront of image recognition tasks. Compared to more recent CNNs, its architecture is fairly straightforward, consisting of five convolution and max pooling layers followed by three fully connected layers.

The next CNN, VGGNet [52], adopted a similar architecture with a few key differences. Firstly, the filter size used for convolution was significantly smaller (3×3 for VGG versus 11×11 for AlexNet). However, the filters typically had a higher number of channels. As the spatial size of the input volumes at each layer decreased (due to convolution and max pooling), the number of filters increased, thus growing the width of the network. Multiple configurations for VGG were proposed, however we implemented the 16 layer configuration which had the highest accuracy.

GoogLeNet [53] was one of the architectures that strayed from the conventional approach of simply stacking convolution and max pooling layers on top of each other in sequence. GoogLeNet introduced the idea of the inception module. In a standard CNN, the data undergoes either a convolution or max pooling operation (there is also the choice of filter size). The inception module allows performing all these operations in parallel, with each inception module carrying out four convolution (with different filter sizes) and max pooling operations in parallel. This creative structuring of layers (and the use of an average pooling instead of fully connected layer at the end) enabled GoogLeNet to achieve a reduction in the number of parameters without sacrificing accuracy.

The final CNN that we investigate, ResNet [54], took the CNN architecture one step further with a few key changes. Instead of widening the network (with a large number of filter channels), its depth was increased, which enabled it to achieve a significantly deeper network without incurring a sizable penalty in the number of increased parameters. It also introduced the idea of a residual block which, instead of simply transforming the input, calculates the term which needs to be added to the input in order to carry out the transformation. We implemented the 50 layer configuration because it represents a good balance between accuracy, memory requirement and throughput.

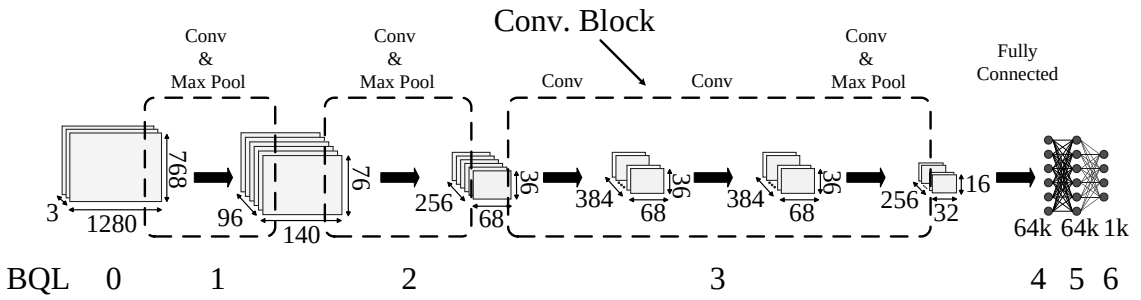


Figure 6 Organization of neural network into block quantized layers (BQL), AlexNet shown as example. Dotted lines represent convolution blocks

Consider a schematic representation for AlexNet in Figure 6. For AlexNet and all other networks, memory and computation (number of floating point operations) requirements are scaled considering the input image dimensions of 1280 × 768. We propose to quantize the feature extraction layers in terms of convolution blocks [55]. Figure 7 shows the size of the output state with each layer showing the data volume that must be transmitted when only a part of the network is computed at the sensor. A reduced data

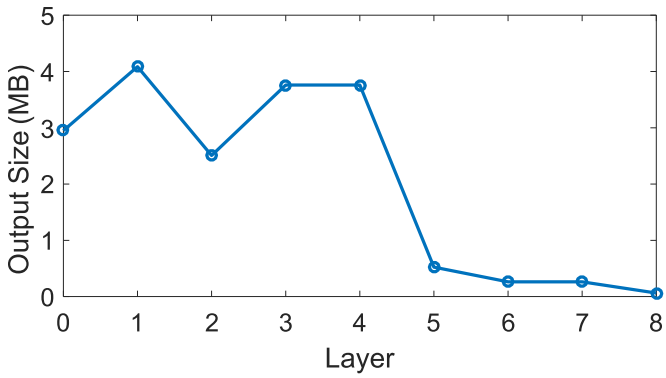


Figure 7 Output size against NN depth for AlexNet

volume helps decrease transmission latency and energy. Initially, the output size increases due to convolution with a large number of filter channels. However, subsequent max-pooling operations decrease the output size. Further, layers 3 to 4 do not reduce data volume, but a max-pooling operation at the 5th layer yields a significant reduction in output state size. Therefore, we organize the 3rd, 4th and 5th layers together into a convolutional block. Hence, we introduce the concept of convolution block (CB) that consists of a set of successive convolution layers followed by a max-pooling layer. Therefore, data volume reduces as more convolution blocks are processed at the sensor. We do not quantize the fully connected layers. To reflect the layer quantization concept, we introduce the term Block Quantized Layer (BQL) for a network, where a BQL refers to a CB in the convolution section, and a FC layer in fully-connection section. Table 1 shows the layers, BQL, CB, and number of operations for all the implemented neural networks.

Table 1 Neural Network Parameters

Neural Network	Layers	Convolution Blocks (CB)	Block Quantized Layers (BQL)	Memory Required (MB)	#OPS (GOPS)
AlexNet	8	3	6	3661	7.17
VGGNet	16	6	9	8695	213.43
GoogLeNet	22	6	7	562	24.75
ResNet	50	5	6	1514	51.43

3.3 Neurosensor Configurations

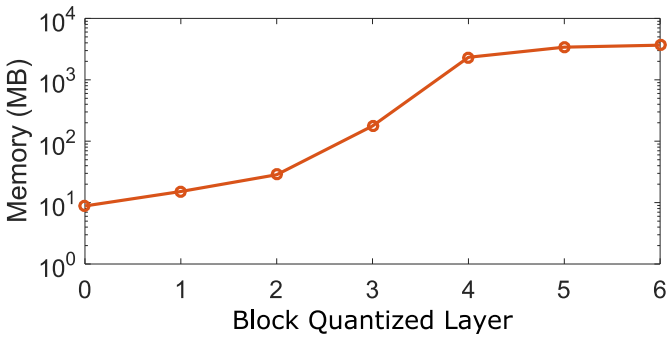


Figure 8 Memory requirements for AlexNet with increasing NN depth

The different hardware configurations for Neurosensor are a direct consequence of the memory requirements of the DNN computation (Table 1). Figure 8 shows the cumulative memory requirement for AlexNet versus network depth. As the network grows deeper, more operations must be performed and more parameters (and output of each layer) must be stored. This requires high density, high-bandwidth memory which also offers concurrent memory access, thus allowing the implementation of the parallel architecture we are pursuing. One option that satisfies these requirements is the hybrid memory cube [28], which essentially stacks DRAM dies on top of one another within the Neurosensor [Figure 9(a)]. We will refer to this configuration as C1. Alternatively, we can have off-chip DRAM for storing the parameters, and use a single SRAM tier for temporary storage of image frames and input/output of a layer, as well as caching the parameters for a layer. Considering a single tier of 14 nm SRAM with effective memory density of 14.5 Mb/mm^2 [56], we project a maximum single tier SRAM capacity of 156 MB. This configuration is referred to as C2 and is shown in Figure 9(b).

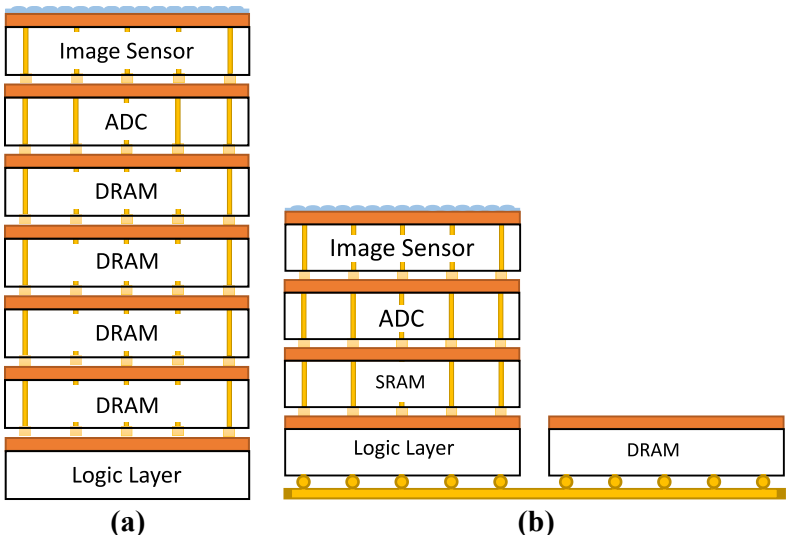


Figure 9 (a) System schematic for C1 with multiple tiers of DRAM (HMC architecture), all weights stored on chip (b) System schematic for C2 with single tier of SRAM, weights stored in off-chip DRAM

3.4 Simulation Framework

Our simulation framework analyzes the Neurosensor to predict power, throughput and sensor noise considering tier-to-tier thermal coupling. Throughout this work, we assume off-chip training where the trained synaptic weights will be overloaded into the memory from HPC. An on-chip deep learning scenario is not suitable for our application as we will not have access to labeled ground truth (on imager output) when the sensor is deployed in field. Hence, all our analysis will be concerned only with inference since the training procedure is performed only once at the very beginning.

3.4.1 System Performance and Power Analysis

The system performance is measured in terms of throughput. The time taken to process an entire frame consists of three components – capture, process, and transmit. The throughput, in terms of frames per second (fps), is given by

$$\text{Throughput} = \frac{1}{t_{capture} + t_{process} + t_{transmit}} \quad (4)$$

where $t_{capture}$, $t_{process}$, and $t_{transmit}$ are the time taken to capture, process, and transmit the image respectively. The capture and processing time are often combined to provide the compute time, and the throughput is then provided by

$$\text{Throughput} = \frac{1}{t_{compute} + t_{transmit}} \quad (5)$$

The total system power (P_{system}) consists of CIS (P_{CIS}) and ADC (P_{ADC}) power, memory (P_{memory}) power, neural logic (P_{logic}) and transmission ($P_{transmit}$) power.

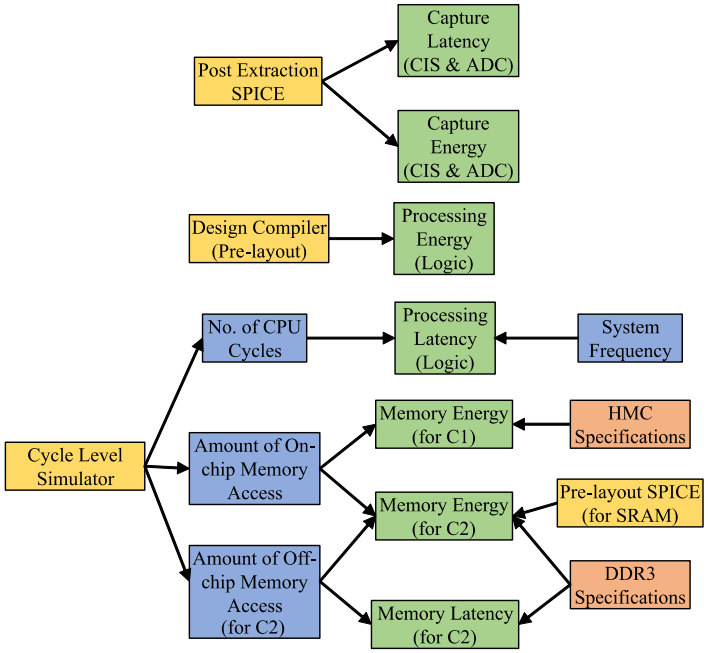


Figure 10 Computation Latency and Energy Calculation Methodology

The capture time for the CIS and ADC are obtained from post-extraction SPICE simulation. To calculate the processing (feature extraction or classification) time, we use a

cycle-level simulator, similar to [27], to find the number of CPU cycles required for computation, and then multiply the number of cycles by the system clock period. Configuration C2 also incurs additional latency because of off-chip memory, which is estimated using the amount of off-chip memory access (from cycle level simulator) along with DDR3 specifications. The capture power and energy (for CIS and ADC) are obtained from post extraction SPICE simulation. Logic energy consumption is acquired from Design Compiler using pre-layout simulation. Memory energy is estimated from either HMC specifications (for C1) or pre-layout SPICE simulation and DDR3 specifications (for on-chip and off-chip memory respectively in C2), coupled with the number of memory access requests (from cycle level simulator). This methodology is summarized in Figure 10.

For the transmitter, transmit time is calculated from the size of processed data and available wireless channel bandwidth, whereas to determine power, different transmitters are assumed for different wireless channel bandwidth conditions and the power is estimated from the relevant datasheets.

3.4.2 Thermal Analysis

Since 3D integration increases the power density of a given system [57, 58], we also perform thermal simulation to study how 3D stacking impacts the system temperature. The thermal simulation framework follows a methodology similar to Lie et. al. [16, 17], and involves solving a 3D RC grid (Fig. 11) using SPICE, where R represents the thermal resistance and C the specific heat capacity associated with each grid. The thermal model

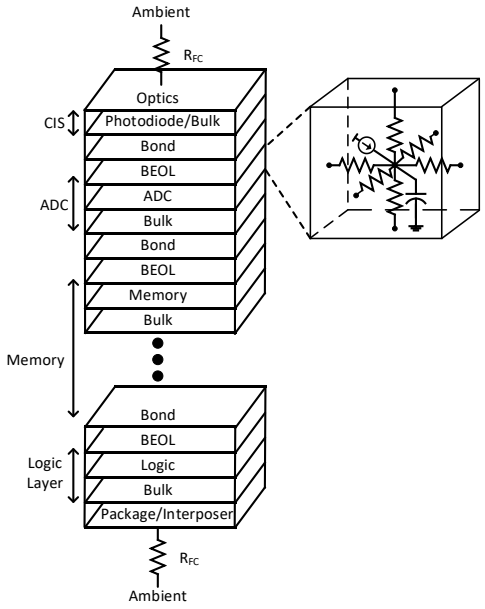


Figure 11 Thermal grid model of 3D-stacked system. The cube represents a grid unit cell. There may be multiple memory layers depending on the configuration.

includes separate tiers for pixel array, ADC, memory (more than one tier depending on configuration), and logic. Floorplan information is also included in the model to simulate the presence of potential hotspots. As the transmitter is off-chip, it is not considered during thermal analysis. Effective thermal conductivity assumes a 1:3 metal to oxide ratio. FEOL consists of silicon and copper and BEOL consists of SiO_2 and aluminum. Termination resistor is derived from the thermal conduction of free convection of air. The stack is covered by microlens, a thick layer of glass with poor thermal conductivity. Bottom layer is a package which consists of a thermal interposer. Table 2 lists the material parameters, obtained from ITRS roadmap [59], used for this analysis.

Table 2 Material parameters for thermal analysis

Parameters	Thickness (m)	R (W/mK)	C (J/m^3K)
Optics	0.001	0.025	3.55M
BEOL	16μ	40	4M
Device Layer	4μ	200	1.75M
Bulk	20μ	200	1.75M
Bond	5μ	0.1	4M
Package	0.001	0.5	3.55M

3.4.3 Noise Analysis

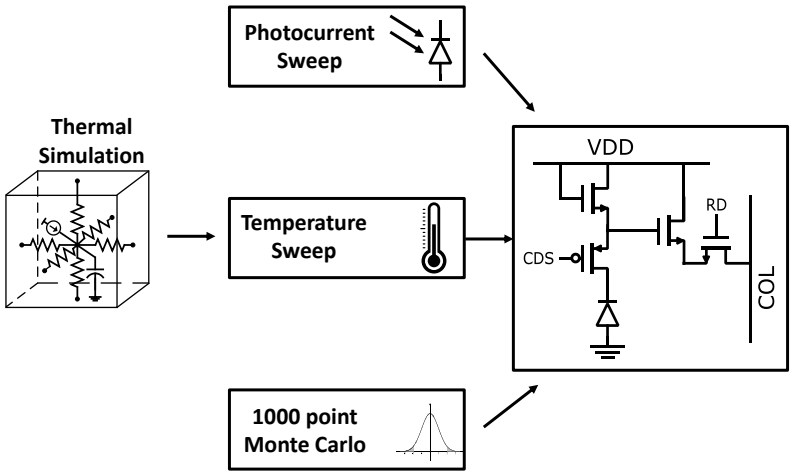


Figure 12 Noise model for CMOS image sensor. Monte Carlo simulation is run at each point of a simultaneous photocurrent and temperature sweep. The temperature values are obtained from thermal simulation

Since pixel response (and fixed pattern noise) is a function of both temperature and illumination [44, 60], we perform coupled thermal simulations and noise analysis (Figure 12). Using SPICE based simulations, we initially sweep the photocurrent from 1pA to 3.7μA in half-decade steps to find out the pixel response to illumination. Next, we repeat this photocurrent sweep for varying temperatures to find the pixel response to different illumination levels at varying temperatures. The temperature points for the sweep are

generated from the results of the thermal analysis. Figure 13 shows the pixel response with different illumination levels and temperature.

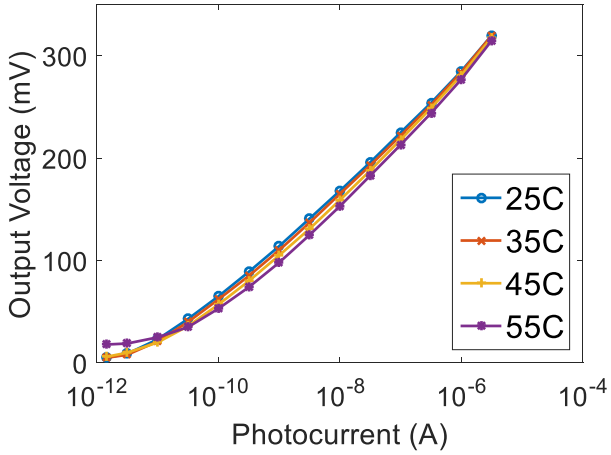


Figure 13 Effect of temperature on imager output

Now that we have a temperature-dependent model for pixel response, this is augmented to include transistor induced noise. To model the fixed pattern noise due to transistor variation, we run SPICE based Monte Carlo analysis using foundry provided variation models. For each temperature-photocurrent pair, 1000 Monte Carlo runs are carried out. Thus this emulates the response over varying photocurrent and temperature of 1000 pixels which suffer from random fixed pattern noise. To simulate the error due to device mismatch at a given photocurrent and temperature, the error values are picked randomly from the 1000 mismatched pixels obtained by Monte Carlo simulation. Figure 14 shows how three sample images change with transistor variation and temperature induced noise for a nominal temperature of 60°C. In general, the image histogram moves towards the left for the high temperature image, which makes the image darker. This also corresponds with the decreased pixel output at elevated temperatures as seen in Figure 13.

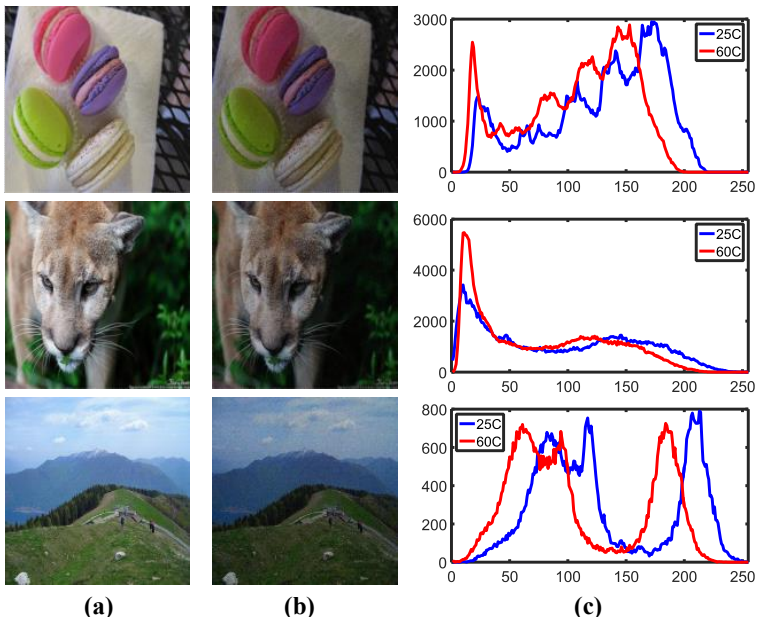


Figure 14 Effect of temperature and transistor induced noise (a) Sample test image (b) Sample image after adding transistor and temperature induced noise at a nominal temperature of 60°C (c) Histogram of images

3.5 Power/Performance Simulation Results

3.5.1 Computation Energy Breakdown

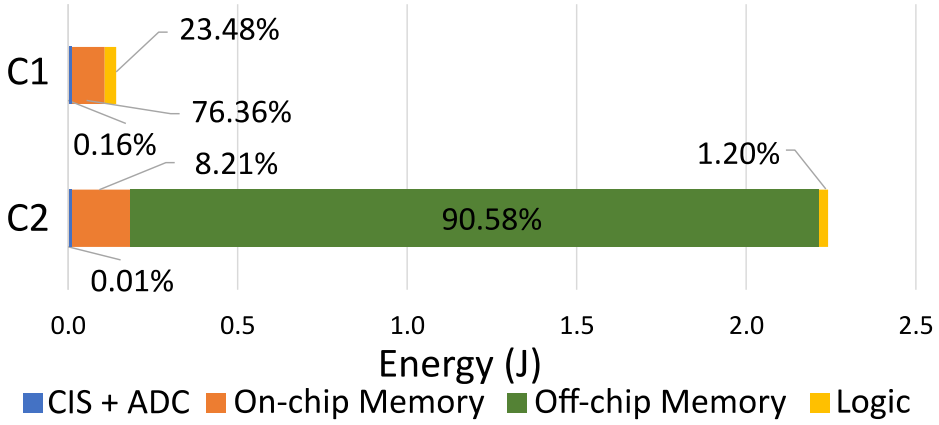


Figure 15 Breakdown of computation energy for configuration C1 and C2 considering complete classification for AlexNet

The power and performance for CIS and ADC are obtained from post extraction simulation, whereas pre-layout SPICE simulations and Design Compiler provide these

metrics for the SRAM and logic layer. The ADC and CIS layer are designed to operate at a frequency of 100MHz, which provides an effective frame capture time of 1.966ms. The logic layer is synthesized to operate at 5GHz, the maximum frequency supported by HMC specifications. Figure 15 shows the various components of computation energy for C1 vs C2 when the full classification pipeline is implemented for AlexNet. Memory energy for C1 is estimated from [28] (3.7 pJ/bit); on-chip memory (SRAM) energy for C2 is obtained from pre-layout SPICE simulation, and off-chip memory energy for C2 is estimated from [61] (70 pJ/bit). For both configurations, memory energy is the dominant component. However, C2 consumes significantly more energy due to off-chip memory access.

3.5.2 System Performance for Configuration C1

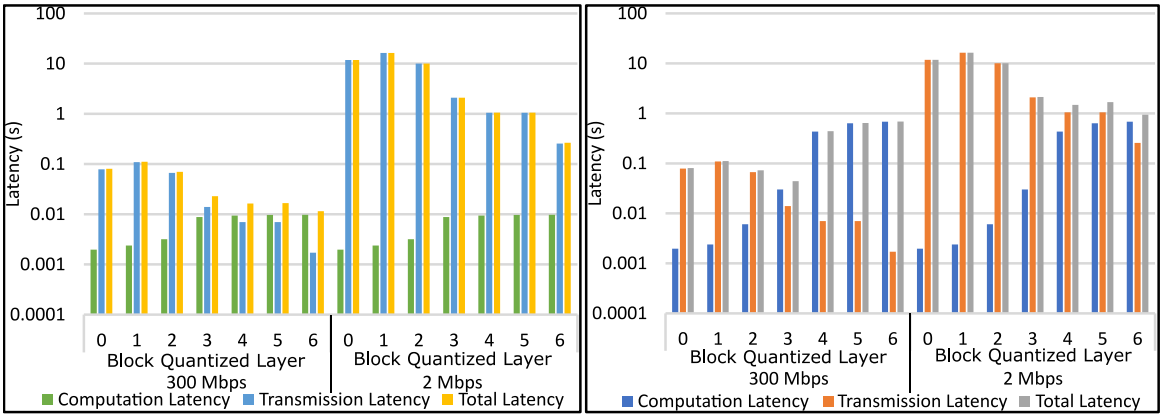


Figure 16 Latency vs NN depth for AlexNet under varying bandwidth conditions for configurations (a) C1 and (b) C2. Computation latency is independent of bandwidth, while Transmission latency varies with bandwidth. Note the sharp increase in computation latency at the fully connected layers for C2 (BQL 4 onwards)

To examine how the system throughput changes as more layers are integrated on the system, we consider computation latency, which depends only on the system frequency, and transmission latency, which is solely determined by host-to-sensor bandwidth, as the two main factors which determine performance. To investigate how the overall throughput

changes with bandwidth, we subject the system to a high bandwidth (300 Mbps) and low bandwidth (2 Mbps) scenario.

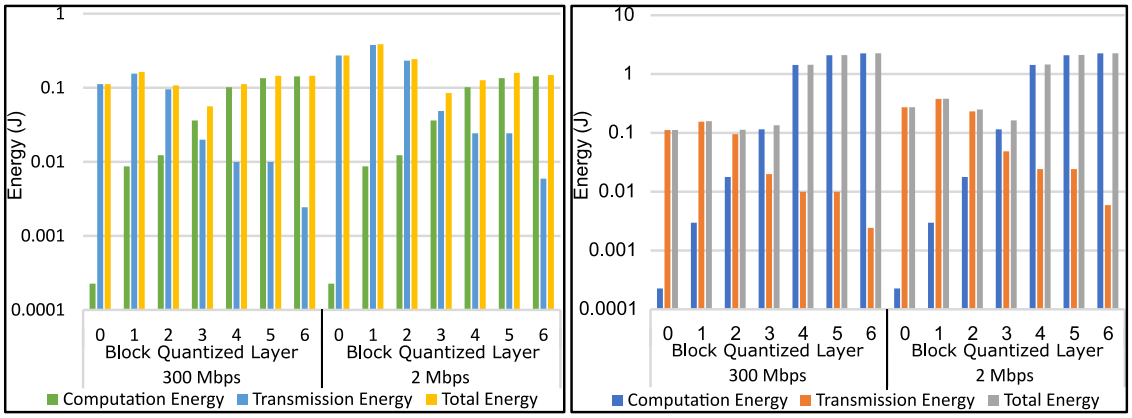
Table 3 Performance Analysis of Neurosensor Configurations

Neural Network	Config.	Wireless Channel Bandwidth (Mbps)	Throughput (FPS)
Baseline	N/A	300	12.4
		2	0.08
AlexNet	C1	300	87.0
		2	3.8
	C2	300	1.5
		2	1.1
VGGNet (VGG16)	C1	300	4.2
		2	2.0
	C2	300	0.6
		2	0.5
GoogLeNet	C1	300	32.7
		2	3.5
	C2	300	9.7
		2	2.8
ResNet (ResNet50)	C1	300	16.8
		2	3.2
	C2	300	3.5
		2	1.9

Figure 16(a) shows compute, transmission and total latencies for Alexnet (with BQL 0 being the input image) for C1. For both high and low bandwidth, the system latency initially increases and gradually decreases as more BQLs are computed in the sensor. As explained in Figure 7, convolution initially increases output size, which creates a bottleneck due to high transmission latency, thus causing an increase (decrease) in overall latency (throughput). However, as more BQLs are computed on-chip, max-pooling operations reduce output size and hence transmission latency. Increased computation time due to more BQL is generally offset by the reduced transmit time, thus decreasing the overall system latency. The effect is more pronounced at lower bandwidth (2 Mbps), where

transmission latency is much larger than computation latency. Table 3 shows the latency for the other neural networks and compares it to baseline latency. For this analysis, we assume the baseline case to be a sensor only system which has neither integrated DNN nor integrated memory, and only transmits the captured image to the host without any processing. In general, irrespective of bandwidth, performing classification on the image and then transmitting the output provides higher throughput for C1. An exception to the above observations is the high bandwidth case for VGGNet, which is so dominated by computation latency that the reduction in transmit time is not enough to offset the increase in processing latency.

3.5.3 System Energy for Configuration C1



(a) (b)

Figure 17 Energy vs NN Depth for AlexNet under varying bandwidth for configurations (a) C1 and (b) C2. Note the large jump in computation energy for the fully connected layers (BQL 4 onwards)

Figure 17(a) shows the energy required in C1 for Alexnet to process and transmit a single frame versus number of BQLs computed in the sensor. The transmission energy for 300 Mbps and 2 Mbps are modeled from datasheets for Ralink (MT7620) router-on-a-chip [62], and NordicSemi transmitter (nRF24L01+) [63] respectively. At the first processing

layer we see an increase in energy because of an increase in the output size, and hence transmission energy. However, as expected from the latency analysis, computing more BQLs in the sensor reduces transmission energy (less data volume) but increases processing energy. In particular, there is a significant increase in the processing energy at the fully connected layers (BQL 4 onwards) due to the large number of weights that have to be fetched from memory. The effect of increased computation energy is more pronounced when wireless channel bandwidth is higher. Hence, processing all layers in the sensor can actually increase energy for 300Mbps bandwidth, but the energy benefit is still observed for 2Mbps case. Table 4 shows the energy consumption for all the networks when entire network is computed on-chip. We observe similar trends to AlexNet with VGGNet being the exception for the low bandwidth case due to its very computation heavy nature.

Table 4 Energy Analysis of Neurosensor Configurations

Neural Network	Config.	Wireless Channel Bandwidth (Mbps)	Energy per Frame (J)
Baseline	N/A	300	0.119
		2	0.279
AlexNet	C1	300	0.145
		2	0.149
	C2	300	2.247
		2	2.251
VGGNet (VGG16)	C1	300	1.089
		2	1.093
	C2	300	5.904
		2	5.907
GoogLeNet	C1	300	0.127
		2	0.131
	C2	300	0.398
		2	0.401
ResNet (ResNet50)	C1	300	0.264
		2	0.268
	C2	300	1.084
		2	1.088

3.5.4 Impact of ADC Architecture on Configuration C1

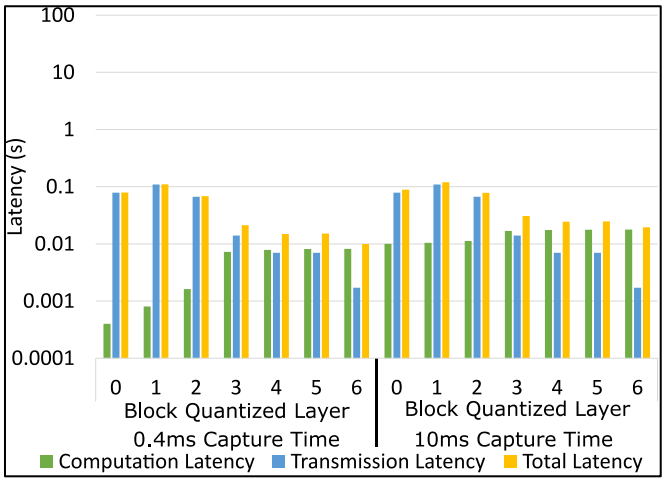


Figure 18 Latency versus NN Depth for AlexNet under varying capture time for configuration C1 (300 Mbps wireless channel bandwidth)

In order to study the influence of ADCs on the performance of our system, we emulate different ADC architectures by varying the capture time. Figure 18 shows how the compute, transmit and total latencies change when different capture times are considered (we consider only the 300 Mbps wireless channel bandwidth for this analysis, since the system is primarily wireless bandwidth-constrained at 2Mbps). Compared to the original of 1.966ms, when the capture time is reduced by 5× to 0.4ms (which emulates a faster ADC architecture), both the compute latency and total latency decreases. However, the trend shown by the original configuration in Figure 16 (a) remains valid – we still get an initial increase in the total latency (because of large wireless transmission latency) followed by a decrease in overall latency as the neural network grows deeper. Switching to a slower ADC, which is modeled by increasing the capture time by 5× to 10ms, causes the total latency to increase. However, the system is still initially bottlenecked by transmission, with the bottleneck shifting to computation as the network grows deeper. Thus, when it comes to performance, different ADC architectures have only the effect of altering the numerical

value of latency, while the trend exhibited with increasing neural network depth remains unchanged. This is mainly because the only contribution of ADC architecture to the computation time is the addition of a capture time, which is independent of the number of layers being integrated on chip.

When it comes to the energy contribution of ADC, we found it to be quite minimal compared to the total energy required for computation, as shown in Figure 15. For configuration C1 considering AlexNet complete classification, for example, the ADC energy is 220 μJ, which is only 0.16% of the total computation energy (142 mJ). Therefore the ADC architecture is unlikely to play any significant role on system energy consumption.

3.5.5 Performance and Energy for Configuration C2

Next, we turn to configuration C2 to investigate the impact of off-chip memory on our system. For this configuration, all the synaptic weights are stored in off-chip DDR3 memory. The weights are transferred to Neurosensor through 16 parallelly operating 32-bit buses. However, using off-chip memory entails a significant latency overhead which is not encountered with configuration C1. Assuming a latency of 12 ns [61] associated with each 32-bit fetch from memory, Figure 16 (b) shows that due to the extra latency associated with fetching weights, the computation latency for C2 is higher than C1. The limitations of using off-chip memory becomes particularly apparent when we reach the fully connected layers (BQL 4 onwards), and involves a significant increase (decrease) in latency (throughput). Due to this increase in computation latency, the system throughput at 300 Mbps wireless channel bandwidth will be lower than the baseline. For restricted

bandwidth, however, the reduction in transmission latency for fully classified image still outweighs computation latency, and a throughput benefit is observed compared to the baseline. Table 3 shows that a similar trend is exhibited by all the other neural networks.

When it comes to energy consumption, because the off-chip memory in C2 consumes significantly more energy compared to C1 (70pJ/bit [61] vs 3.7pJ/bit) , C2 is considerably more energy hungry, especially when the fully connected layers are implemented, as seen in Figure 17 (b). Because of this computation energy increase, implementing the entire network and transmitting the fully classified image for C2 always consumes higher energy than the baseline irrespective of bandwidth. Table 4 shows that all the other networks echo this trend as well.

In summary, computing the entire DNN in the sensor improves energy-efficiency only when memory is integrated in 3D with the sensor. It should be noted that we do not analyze the impact of ADC architecture on configuration C2 because both the capture latency and energy are insignificant components of computation latency and energy respectively, and hence are unlikely to have any major effect on computation metrics.

3.5.6 Optimal Energy Efficiency

So far we have examined four possible configurations in terms of hardware and the available bandwidth between sensor and host, and investigated how the energy and performance vary as we implement the neural networks layer by layer. The preceding discussion shows that, with a 3D integrated memory (configuration C1), processing the entire DNN in the sensor improves performance. However, energy is minimized when an optimal number layers are processed on-chip, and beyond this layer, the energy efficiency

decreases. The partitioned inference (Figure 19) concept is therefore introduced which aims to achieve optimal energy efficiency by implementing only part of the DNN classification pipeline on the sensor side, and performing the rest of the computations on the sensor side through the transmission of intermediate features. It should be mentioned that this concept of partitioned inference was developed in collaboration with Dr. J. H. Ko, and is also explored in his PhD dissertation [64]. However, [64] assumes a conventional sensor architecture, and investigates the system implications through an algorithmic approach by exploring weight compression, and retraining the DNN engine on the sensor side to mitigate accuracy loss due to compression. This work, on the other hand, leaves the DNN pipeline as it is, and the main focus is to study how the sensor architecture itself can determine the optimum energy efficiency configuration.

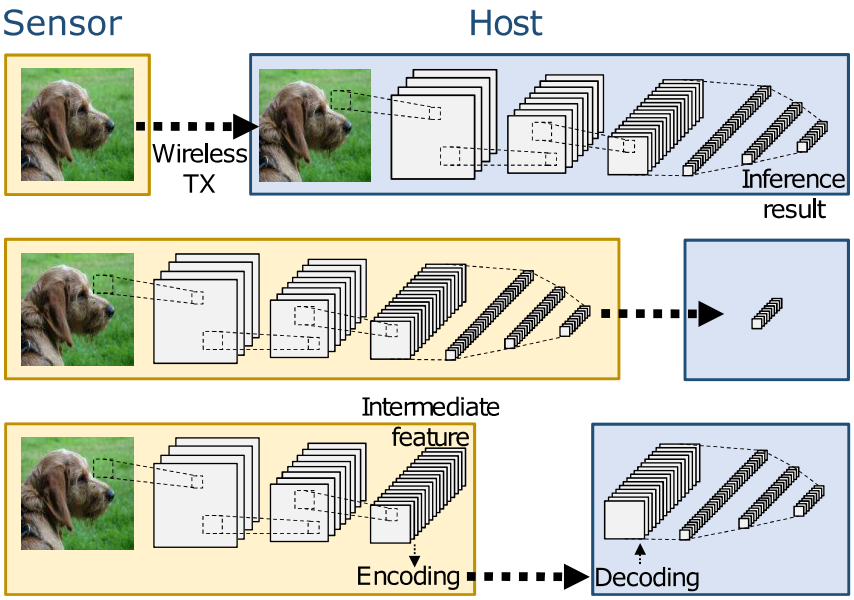


Figure 19 Performing classification entirely on the host entails large transmission overhead, whereas implementing the DNN completely on the sensor side involves large computation energy. Partitioned inference [64] allows trade-offs between transmission and energy overhead to achieve optimum energy efficiency by partitioning the DNN pipeline between the sensor and host, and transmitting only the intermediate features.

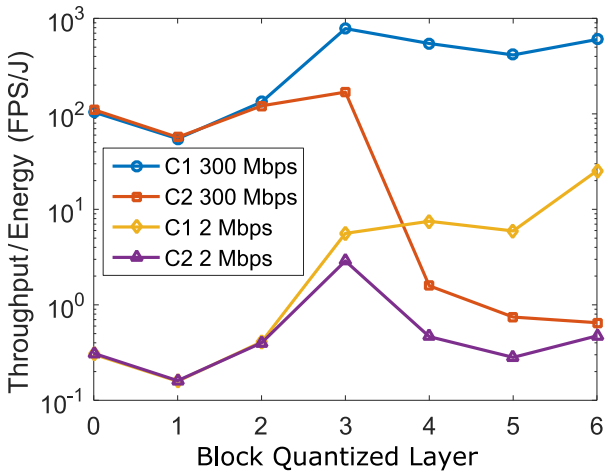


Figure 20 Throughput to Energy Ratio (TE ratio) vs Neural Network Depth for AlexNet considering different configurations under varying bandwidth. Higher TE Ratio represents better energy efficiency.

To identify the optimal depth of DNN processing to achieve maximum energy efficiency, we use the throughput to energy (TE) ratio, measured in FPS/J, as a metric. The higher the ratio, the better the energy efficiency. Figure 20 shows how the energy efficiency of AlexNet changes versus BQLs. In general, low wireless channel bandwidth yields lower energy efficiency due to increased transmission latency and energy. Configuration C2 is less energy efficient than C1 because of higher computation energy and latency associated with off-chip memory access. As the data moves through the network, the TE ratio initially drops because of an increase in output size, which reduces throughput and increases energy, but as more layers are processed in the sensor, the energy-efficiency increases due to reduced output data volume. However, when the fully connected layers are reached (BQL 4 onwards), the energy-efficiency tends to drop due to the large increase in computation energy associated with fully connected layers. In AlexNet, we observe that implementing only the convolutional layers in the sensor provides maximum energy-efficiency with higher bandwidth wireless channel and/or configuration C2 (off-chip DRAM). However,

in C1 under 2 Mbps channel, the energy-efficiency continues to improve even when fully-connected layers are processed because the increase in throughput (and reduction in transmission energy) achieved by moving to the fully connected layers outweighs the increased computation energy consumption.

Table 5 DNN processing for Optimum Energy-efficiency

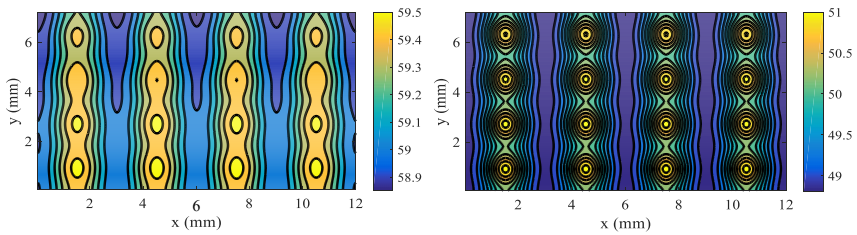
Neural Network	Config.	Wireless Channel Bandwidth (Mbps)	# of BQLs (Optimum/ Total)	Throughput (FPS)	Energy per Frame (J)	Gain in FPS/J over Baseline
Baseline	N/A	300	N/A	12.4	0.119	N/A
		2	N/A	0.08	0.279	N/A
AlexNet	C1	300	3/6	43.9	0.056	7.52
		2	6/6	3.8	0.149	88.94
	C2	300	3/6	22.8	0.135	1.62
		2	3/6	0.5	0.163	10.70
VGGNet (VGG16)	C1	300	6/9	3.7	0.917	0.04
		2	9/9	2.0	1.093	6.38
	C2	300	4/9	2.2	0.988	0.02
		2	6/9	0.2	1.527	0.46
GoogLeNet	C1	300	6/7	32.7	0.125	2.51
		2	7/7	3.5	0.131	93.18
	C2	300	6/7	11.0	0.358	0.29
		2	6/7	2.8	0.3617	26.99
ResNet (ResNet50)	C1	300	6/6	16.8	0.264	0.61
		2	6/6	3.2	0.268	41.64
	C2	300	3/6	9.3	0.258	0.35
		2	6/6	1.9	1.088	6.09

Table 5 summarizes the performance and energy at the optimal energy-efficiency point for different networks, and also compares the optimal energy-efficiency with that of the baseline case. First, we observe that for ResNet, optimal condition occurs when all layers are processed in the sensor. This is because ResNet uses an average pooling layer near the end to obtain a confidence of categories, thus bypassing cascaded fully connected layers and minimizing the associated energy penalties (its single fully connected layer also involves a reduction in output size, further increasing the throughput/energy ratio). Second,

we observe that energy-efficiency gain is more pronounced for 2 Mbps wireless channel (transmission is relatively more expensive) and C1 configuration (computation latency and energy are lower due to 3D architecture). Finally, we note that for several cases, for example VGGNet, even optimal energy-efficiency is worse than the baseline case, mostly due to the large computation requirements.

3.6 Thermal Simulation & Noise Analysis Results

3.6.1 Thermal Analysis



(a) (b)

Frequency (GHz)	CIS TEMP. (°C)	
	C1	C2
1	32	30
2	39	35
3	45	40
4	52	45
5	59	51

(c)

Figure 21 Temperature (°C) of the CIS layer for a nominal operating frequency of 5 GHz for (a) C1 and (b) C2. (c) shows how the temperature varies with operating frequency (25°C operating temperature)

Thermal simulation is performed using the methodology discussed in Section 3.4.2. Considering front and back-end layers, we have a total of 27 tiers of material for C1 and 14 tiers for C2 in the thermal stack. The main impact of the thermal issues will be the introduction of noise on the image sensor output, so our investigation will be focused on the CIS layer. Figure 21 shows the temperature map of the image sensor layer for a nominal system operating frequency of 5 GHz, considering 25°C ambient temperature. The image sensor layer for configuration C2 will always exhibit lower temperature than C1 because of lower power consumption for the sensor die-stack (we exclude the off-chip memory power in C1 for thermal analysis because this is outside the thermal stack). We repeat our

thermal simulation to find the temperature map for operating frequencies from 1 GHz to 5 GHz, the maximum operating frequency supported by HMC specifications. Do note that since the temperature differential between the hottest and coldest points on the CIS layer are quite low, we will be dealing with average CIS temperature from now onwards. Figure 21 (c) shows that the average CIS tier temperature increases linearly as the frequency of operation increases. The maximum temperature attained is 59°C, which is well within the thermal limits (105°C) allowed by HMC specifications. We will use these values to inject temperature induced noise into test images, and examine the effect this noise has on neural network classification accuracy.

3.6.2 Effect of Noise on Neural Network Accuracy

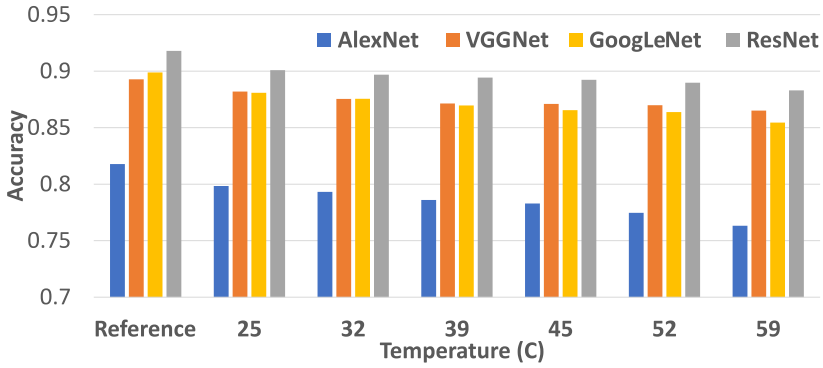


Figure 22 Impact of temperature and transistor induced noise on top-5 accuracy of CNNs

We follow the methodology discussed in Section 3.4.3 to estimate noise at CIS considering temperature and transistor variations. As C1 has a higher temperature than C2, we will be performing our noise simulations only for configuration C1. Since our developed noise model is able to account for both temperature and transistor induced variation, we run the noise model at each temperature point obtained from thermal simulation. This provides us with a set of 1000 noisy pixels at each temperature and photocurrent, which we can then

use to inject random noise into our test sample of 10000 images from the ImageNet dataset. The end result is a set of noisy images at 6 different temperatures, 32°C, 39°C, 45°C, 52°C, 59°C and 25°C, with each temperature containing 10000 noisy images (we add the 25°C temperature to examine the effect of transistor variation by itself without temperature induced noise).

Table 6 Neural Network Throughput/Accuracy Trade-off

Neural Network	Frequency (GHz)	CIS Temp. (°C)	Accuracy	Throughput (FPS)	
				300 Mbps	2 Mbps
AlexNet	1	32	0.793	23.456	3.368
	5	60	0.763	87.023	3.763
VGGNet	1	32	0.875	0.859	0.705
	5	60	0.865	4.243	2.041
GoogLeNet	1	32	0.876	7.235	2.548
	5	60	0.854	32.701	3.510
ResNet	1	32	0.897	3.531	1.860
	5	60	0.883	16.784	3.186

We run inference on test image samples using pre-trained Caffe [65] models. Figure 22 shows how the top-5 accuracy changes with temperature. The reference column represents the accuracy for ideal software simulation (without any temperature or transistor induced noise). In general, the accuracy decreases at higher temperature because of increased temperature induced noise. AlexNet has the lowest accuracy among the four networks, and is the one most prone to noise. AlexNet suffers a 6% accuracy drop, from 82% for the reference images to 76% at 59°C. ResNet has the highest accuracy and maintains a low error rate (88.3% top-5 accuracy) even under the highest temperature. VGGNet is seen to be the most noise tolerant network, with the accuracy dropping by only 2.8%. This resilience to noise causes it to surpass GoogLeNet at higher temperatures, which suffers an accuracy drop of 4.5%.

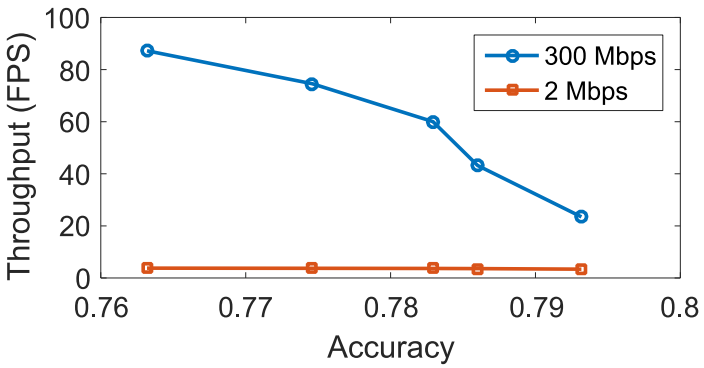


Figure 23 Top-5 Accuracy vs throughput for AlexNet considering varying bandwidth for configuration C1

Figure 23 shows how the classification accuracy of AlexNet changes with varying throughput. A higher throughput requires higher operating frequency, which in turn increases temperature and noise in the CIS, resulting in reduced classification accuracy. The trade-off is more pronounced for the 300Mbps case as the system throughput strongly depends on computation latency (determined by clock frequency). For the low bandwidth condition, higher accuracy can be achieved with a relatively low penalty to throughput. Table 6 shows this tradeoff for the other networks considering the highest and lowest operating frequency. Do note that the above conclusion does not apply to VGGNet because its throughput is dominated by computation latency irrespective of bandwidth between sensor and host.

3.7 DNN Architecture Dependency

In this section we discuss how the neural network architecture can influence the performance and energy, and hence optimum energy configuration for a particular network. Of the four neural networks that we study, AlexNet and VGGNet have a more conventional architecture where convolution and max-pooling layers are arranged alternately, followed by three fully connected layers. GoogLeNet modifies the standard architecture through the

use of inception modules, while ResNet goes for a deep architecture instead of widening the neural network. Both GoogLeNet and ResNet forego the traditional approach of using cascaded fully connected layers, and use only one fully connected layer near the end. As Figure 8 shows, cascaded fully connected layers typically involve a large jump in memory requirement and entails a significant amount of memory access, which causes in turn performance and energy penalties (this is more apparent for configuration C2 due to off-chip memory access). Innovations at the DNN architecture level can serve to mitigate these penalties. For example, despite being a deeper network, GoogLeNet consumes lower energy than AlexNet and VGGNet primarily because of its low memory requirement (Table 1). In fact, for configuration C2 where off-chip memory access is the key parameter for determining throughput, low memory requirement (and the resultant reduction in memory access) causes both GoogLeNet and ResNet to achieve higher throughput than AlexNet despite being more computation heavy (24.75GOPS for GoogLeNet, 51.43GOPS for ResNet, vs. 7.17GOPS for AlexNet). The impact of DNN architecture towards optimum energy efficient configuration can also be observed in Table 5. For configurations where computation is the primary bottleneck (high wireless channel bandwidth for configuration C1, and all bandwidths for configuration C2), the optimum energy efficiency point lies just before the fully connected layer for AlexNet, VGGNet and GoogLeNet. However, ResNet typically provides optimum energy efficiency when the full network is implemented because its single fully connected layer involves only a minor energy penalty while offering a large reduction in output size, leading to lower transmission latency and energy (do note that despite GoogLeNet having a single fully connected layer as well, its FC layer involves only a minor reduction in output size; thus the GoogLeNet optimum configuration does not

include the fully connected layer). To summarize, the choice of neural network, and its architecture, can have a pronounced effect towards determining performance/energy considerations as well as energy efficiency of Neurosensor. For configurations using 3D-stacked on-chip memory where memory access latency is not an issue (C1), DNN architectures with a reduced number of operations provide higher system throughput; computation latency is relatively independent of DNN memory requirements for this configuration. However, when the memory is implemented off-chip (C2), memory access becomes the principal component of computation latency. For this configuration, DNN architectures with lower memory requirements provide increased throughput despite requiring a higher number of operations.

3.8 Comparison with Prior Neurosensor Design

A preliminary version of this system was presented in [66]. The main difference between the two works relates to the digital portion of Neurosensor. Since the imager (sensor and ADC), as well as 3D DRAM dies (for HMC configuration) are identical for both the works, all associated parameters for imager and HMC are identical. The neural logic tier (and SRAM tier, where applicable) has been scaled from 28nm. As a result of this scaling, the on-chip SRAM capacity (for SRAM configuration) has increased from 62MB (assuming 1.39mm^2/MB density [67]) to 156MB (assuming 14.5Mb/mm^2 density [56]), and the maximum operating frequency has increased from 300MHz to 5GHz. Even though the logic layer is the same architecturally for both the works, the significantly increased operating frequency has resulted in a large throughput increase. However, this higher operating frequency entails increased power consumption, and subsequently, imparts thermal concerns which were not encountered in the previous work. Table 7

compares some of the key parameters between the two works. Note that the table only compares between the HMC configurations as the SRAM configuration in [66] did not include off-chip DRAM.

Table 7 Comparison with Prior Neurosensor Design

		[66]	This Work
System Footprint		12mm × 7.2mm	
Sensor Resolution		1280 × 768	
Pixel Size		9µm × 9µm	
Sensor & ADC Feature Size		130nm	
Sensor & ADC Power		115mW	
Logic Layer Feature Size		28nm	15nm
Maximum Operating Frequency		300MHz	5GHz
Memory (HMC) Power		568mW	9.47W
Logic Power		240mW	3.41W
Computation Energy for Single Frame		155mJ	142mJ
Maximum Throughput for AlexNet	300 Mbps Wireless Channel	7.49fps	87.0fps
	2 Mbps Wireless Channel	2.58fps	3.80fps
Average Sensor Temperature (maximum operating frequency)		27.26°C	59°C

3.9 Summary

This chapter explored the design of Neurosensor - a 3D stacked image sensor with integrated logic for deep neural network computation. Our analysis showed the potential energy-efficiency advantage of integrating DNN computation in the sensor. However, the improvement depends on system architecture and target application, and innovations in DNN architecture can be leveraged to achieve performance and energy improvements. We observe that configuration C1 with the memory for parameter storage integrated within the 3D sensor shows much higher throughput. Likewise, in-sensor processing is more advantageous for bandwidth-constrained wireless channels. Moreover, we observe that instead of implementing the entire DNN on chip, it is often more advantageous in terms of

energy efficiency (measured in terms of FPS/J) to implement only the feature extraction layers on chip and offload the classification computations to the host. The coupled power, thermal, and noise analysis shows degraded accuracy at high system throughout (frequency and power) because of elevated temperature (noise). In summary, Neurosensor presents an image sensor architecture that captures image, performs on-site neural computations, and explores tradeoffs among system frequency, wireless channel bandwidth, and image classification accuracy.

For the next part of this work, we will explore how we can further utilize 3D integration to increase parallelism at the image sensor level, and explore processor-in-memory computing through the use of emerging devices.

CHAPTER 4. ENHANCING ENERGY EFFICIENCY THROUGH PIXEL-LEVEL PARALLELISM AND PROCESSING IN MEMORY COMPUTING

4.1 Pixel-level parallelism using digital pixels

The discussion in the previous chapter has demonstrated how 3D integration can be utilized to implement sensor-integrated computation architectures. While we have seen that 3D integration of the processing and memory layers can be leveraged to realize highly parallel energy efficient neural accelerators, our sensing mechanism still uses the conventional approach of row-by-row scanning, which imposes limitations on the capture time and does not fully achieve the full parallelism potential brought forth by the 3D structure. In the following discussion, we investigate a new class of 3D-integrated digital pixels which effectively integrate a simplified ADC for each pixel, and employs massive level parallelism by A/D converting all the pixels simultaneously – thus leading to significant reduction in sensor capture time.

4.1.1 Digital Pixel Architecture

Since each pixel in our 3D-integrated digital pixel contains its own dedicated ADC, the use of a conventional ADC which matches the pixel pitch is not feasible (e.g. our pixel dimensions are 9μm×9μm whereas the ADC size in CHAPTER 3 is 120μm×50μm). To match the constraints put forward by an in-pixel ADC architecture, we use a PFM-ADC [20] architecture. The high-level overview of the digital pixel can be seen in Figure 24. The photocurrent to frequency converter (PFC) generates pulses at a rate which is dependent

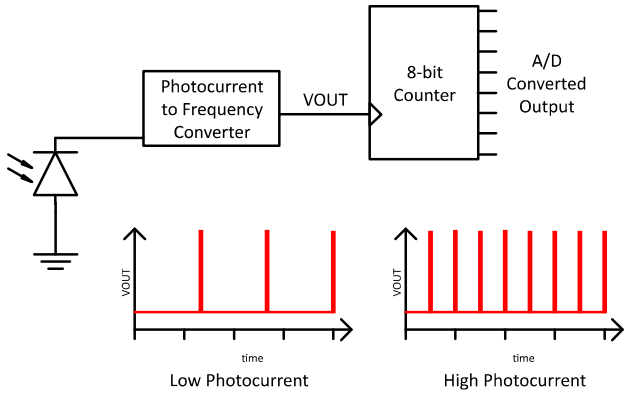


Figure 24 Overview of PFM-ADC operation of 3D Integrated Digital Pixel

on the photocurrent being generated by the photodiode. The output of the PFC is applied to the clock input of an 8-bit counter. Thus, for a given capture time, this architecture performs A/D conversion by counting the number of pulses produced by the PFC, which is proportional to the photocurrent (and illumination).

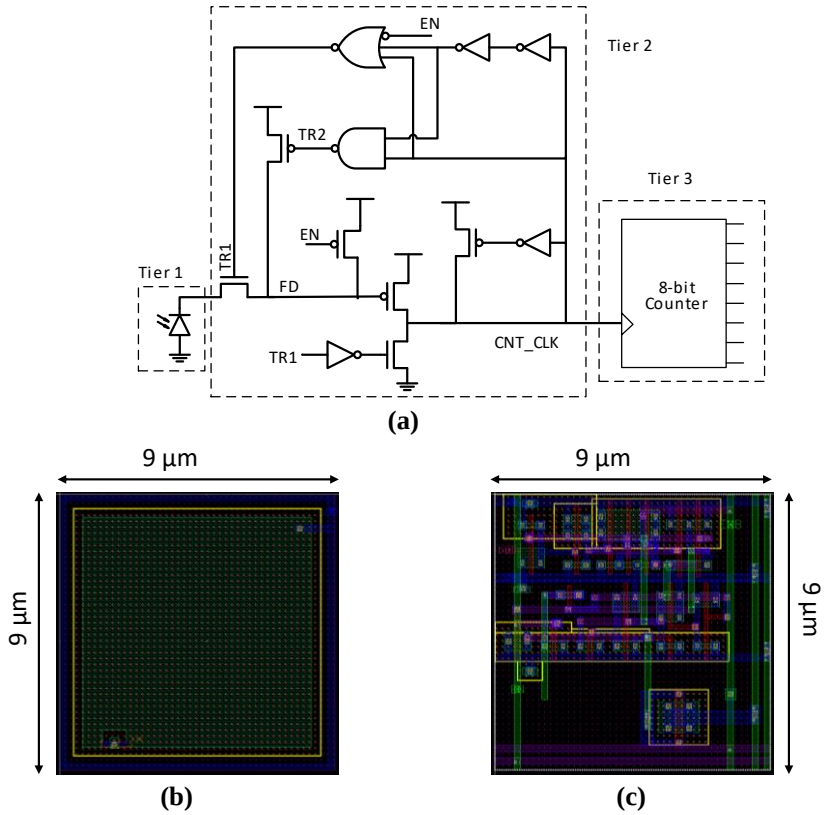


Figure 25 (a) Circuit schematic of digital pixel (b) Photodiode layout (c) PFC layout

Figure 25(a) shows the circuit schematic of the digital pixel, and the layout of a single pixel in the photodiode tier and PFC can be seen in Figure 25 (b) and (c). The pixel footprint is the same as that introduced in Neurosensor (9µm×9µm). However, to accommodate all the additional A/D conversion circuitry for each layer, we move to a 3-tier structure for the sensor (as opposed to 2-tier in the previous configuration), with tier 1 being dedicated entirely to the photodiodes, tier 2 containing the PFC, and tier 3 containing the 8-bit counters. The tiers communicate to each other through Cu-Cu connections, similar to [21], which allows the placement of dense 3D interconnections.

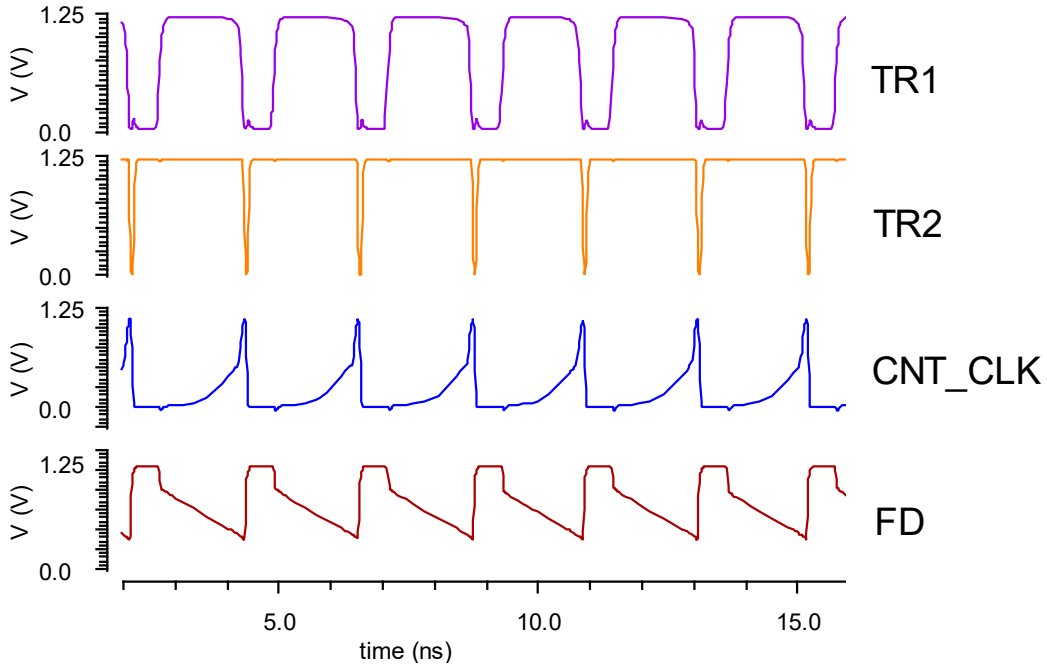


Figure 26 Simulation waveform of digital pixel

Figure 26 provides a simulation waveform of the typical operation of the pixel. The EN signal can be used to control the frame rate (how often a frame is captured) by turning

off the entire pixel between successive sampling events. The CNT_CLK signal can be seen to be producing a train of pulses, which is applied to the CLK input of an 8-bit counter, thus realizing 8-bit A/D conversion.

4.1.2 Pixel Response and Noise Characteristics

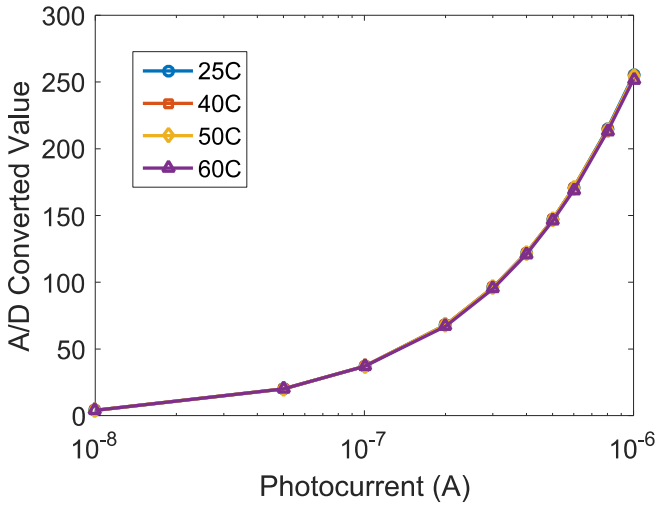


Figure 27 Digital pixel response against photocurrent over varying temperature

Figure 27 shows how the digital pixel output changes as the photocurrent is varied. A temperature sweep was also carried out to investigate the dependence of pixel output on temperature. However, as seen from the figure, the digital pixel is quite resilient to changes in temperature, and the photocurrent response at varying temperatures practically overlap one another.

A methodology similar to that introduced in 3.4.3 was used to add transistor and temperature induced noise to the image. Figure 28 shows the original test image as well as the noisy image at a nominal temperature of 60°C. Since the pixel response is quite insensitive to temperature, the histograms show that the noisy image at high temperature

is virtually identical to the test image, with the only difference between the two being the addition of transistor induced noise.

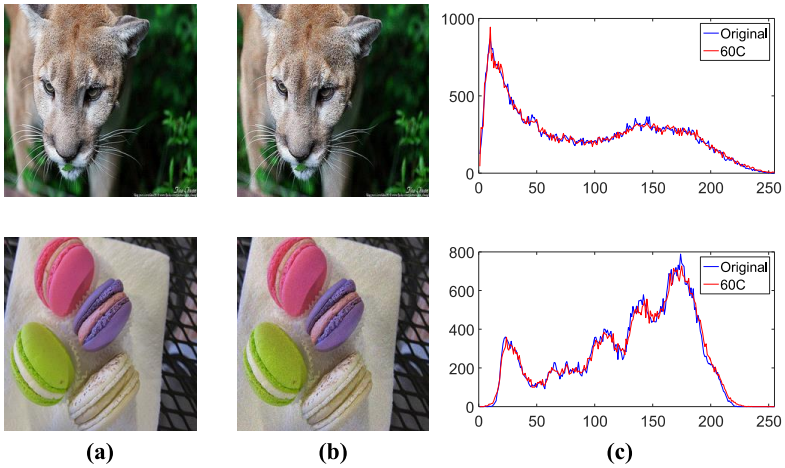


Figure 28 (a) Test image (b) Test image with transistor induced noise at 60°C (c) Histogram of images

4.1.3 Sensor Power and Throughput

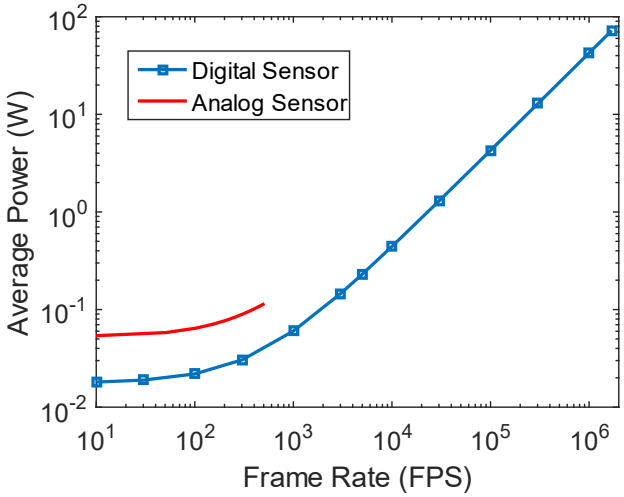


Figure 29 Frame rate versus power consumption for digital sensor versus analog sensor

The principal advantage of using digital 3D-integrated pixels is the significant boost in sensor throughput due to the massive level parallelism since all pixels are undergoing

A/D conversion simultaneously. Running at maximum speed, the digital pixel sensor can convert an image every 590ns, which provides an effective frame rate of 1.69×10^6 frames/second. However, integrating ADC into a pixel also causes a marked increase in power draw since the total number of ADCs in the sensor array increases by several orders. Assuming the sensor resolution remains the same as that presented in earlier works, a 1280×768 pixel array, running at maximum speed, would consume 72.1W of power. Running the sensor array at the maximum possible speed is thus not feasible because of the large power requirements, and also because designing a system capable of storing images at such a high data rate will not be trivial. However, since the pixels are mostly digital in nature, we have the option of scaling down the frame rate, and achieving a proportional reduction in power draw. Interestingly, if the digital sensor is operated at frame rates similar to that as the analog sensor, we actually observe savings in power due to the absence of analog bias currents (Figure 29).

Throughout this discussion, we have seen how 3D integration can be combined with in-pixel ADCs to yield highly parallel, high throughput imagers. However, our preceding discussion in the previous chapters has shown that for sensor based neural acceleration framework, it is often the processing portion that is responsible for the principal bottleneck. Therefore, in the next section, we are going to couple this imager with emerging device based processor in memory architectures to take full advantage of the high throughput brought forward by the digital pixel based sensor array.

4.2 Processing-in-Memory Architecture using ReRAM

Processing-in-Memory (PIM) architectures offer the potential to integrate

computation and storage within a memory device, thus eliminating the separation between compute and data, and leading to advantages in terms of throughput and energy efficiency [68-70].

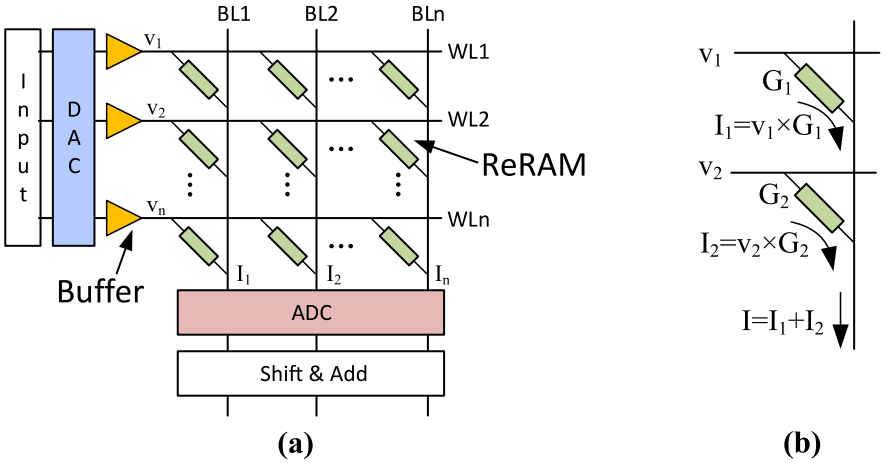


Figure 30 (a) ReRAM crossbar based PIM architecture (b) Multiply-Accumulate operation using memristors

Recent efforts have focused on using non-volatile memory (NVM), particularly resistive RAM (ReRAM) architectures to perform in-memory computation [33-35, 71]. Simply put, an ReRAM device consists of a resistive switching layer (e.g. HfO_x, NiO, TiO_2, Al_2O_3) sandwiched between two electrodes (e.g. Pt, TiN) [72, 73]. The resistance of an ReRAM device is variable, and can be modulated by applying a set or a reset voltage. In ReRAM PIM architectures, the principal idea is to utilize a crossbar array to carry out vector matrix multiplication (VMM) [74, 75] using mixed signal computation. Figure 30(a) shows a basic ReRAM based VMM unit with crossbar and peripherals, and Figure 30(b) shows how ReRAMs can be used to carry out multiply-accumulate operations. The conductance of the memristors can be programmed to the values G_1, G_2,…G_n, and so on. If the voltages V_1,V_2,…V_n are applied to each of the n rows, cell i passes current $I_i = V_i \times G_i$ into the bitline according to Kirchoff's law. As Figure 30(b) shows, the total current into

the bitline is the sum of currents passed by each cell in the column. Thus this current I represents a dot product vector operation between the set of input voltages at each row (V) and the set of conductances at in a given column (G).

The neural network parameters are programmed into the device conductances, and the input vectors are applied as analog wordline voltages. Therefore the current emerging from each bitline can represent the neuron outputs in a CNN, where each neuron is subjected to the same inputs, but produces different outputs due to a different set of synaptic weights. Since the multiplication operation in ReRAM array is primarily analog, the digital inputs into the array must be first converted to analog signals through a DAC and the analog outputs must be converted back into digital through an ADC. This digital conversion is necessary in order to communicate with the other digital blocks in the system.

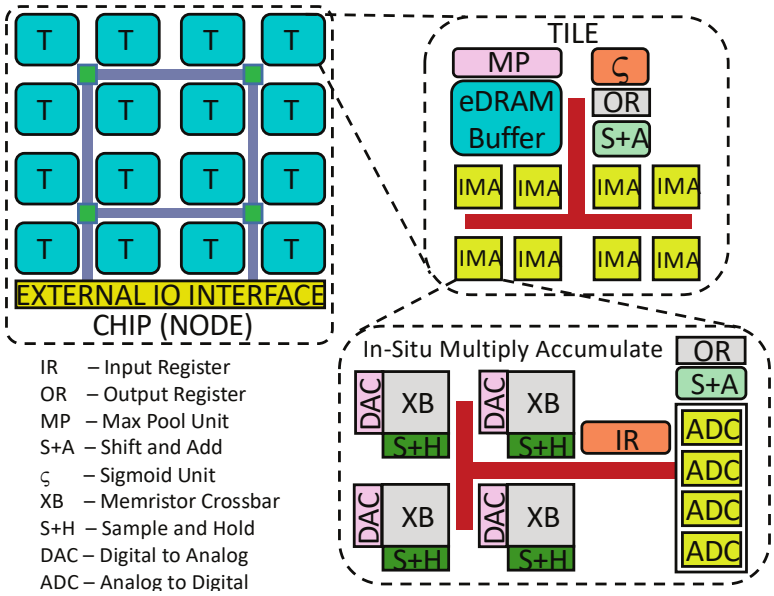


Figure 31 ISSAC architecture hierarchy [33]

In order to investigate the system level impact of coupling highly parallel 3D integrated sensors with PIM architectures, we evaluate a previously published, well-known

ReRAM based neural accelerator – ISAAC[33]. ISAAC was one of the first works to design and characterize a full-fledged accelerator based on crossbars. The work introduced a pipelined architecture with some crossbars dedicated for each neural network layer and eDRAM buffers that gather the data between the pipeline stages. Figure 31 shows the basic architecture of the system. The entire chip is composed of a number of tiles (T) connected through a concentrated mesh. Each tile contains eDRAM buffers to store input values, in-situ multiply-accumulate (IMA) units, and output registers to combine results. The tiles also contains shift-and-add, max-pool and sigmoid blocks. Each IMA contains a number of crossbar arrays and ADCs, as well as input/output registers and shift-and-add units.

During operation, inputs are provided to ISAAC through the external IO interface, which is then directed to the tiles processing the first CNN layer. An FSM inside the tile routs these inputs to the corresponding IMAs. The IMAs carry out the required dot product operations required for feature extraction and classification, and send the results to ADCs, which are combined in the output registers after shift-and-add operations. To reduce power and area overhead due to ADCs, ISAAC shares the ADCs among multiple crossbars. The IMA output is passed through the sigmoid operator, and stored in the eDRAM banks of the tiles processing the next CNN layer. This process continues until the final layer generates an output which is then sent out through the I/O interface. ISAAC also supports the use of multiple chips in order to process larger networks.

The basic architecture in ISAAC has a number of configurable parameters such as (1) the ReRAM crossbar array size, (2) the number of crossbars in IMA, (3) the number of ADCs in IMA and (4) the number of IMAs in a tile. These parameters were tuned in order to optimize three metrics - (1) Computational Efficiency (CE), the number of operations

performed per second per unit area, (2) Power Efficiency (PE), the number of operations performed per second per watt, and (3) Storage Efficiency (SE), the on-chip capacity for storing synaptic weights per unit area. This gives rise to three configurations for ISAAC which optimize performance, energy and storage respectively [33], which are provided below

1. ISAAC-CE is optimized for performance, and contains 12 IMAs per tile, 8 ADCs per IMA, and 8 crossbar arrays of size 128×128 per IMA.
2. ISAAC-PE is optimized for energy, and contains 16 IMAs per tile, 8 ADCs per IMA, and 8 crossbar arrays of size 128×128 per IMA.
3. ISAAC-SE is optimized for storage, and contains 4 IMAs per tile, 8 ADCs per IMA, and 512 crossbar arrays of size 256×256 per IMA.

In the subsequent sections, we will discuss the system level implications of coupling these ReRAM based PIM architectures with the digital sensor pixel, and explore the impact of system architecture on system performance and energy efficiency.

4.3 System Overview

So far on our discussion regarding 3D integrated sensors with deep neural network computations, we have covered two types of sensors. Both the sensors have an identical resolution of 1280 × 768 pixels, as well as identical pixel pitch (9μm × 9μm), but differ in architecture, sensing mechanism and throughput.

1. The first sensor architecture, described in CHAPTER 3, is primarily analog in nature. It consists of two tiers, consisting of pixels in the top tier followed by an array of ADCs

in the tier below it. Despite the 3D structure, the sensor still performs row-by-row readout, and does not fully take advantage of the parallelism offered by 3D integration. Since the ADCs are primarily analog circuits, a large portion of the power consumption for this sensor architecture comes from bias power consumption of the ADC comparators, and the power is relatively insensitive to performance scaling.

2. The second sensor architecture, discussed in Section 4.1, is composed of "digital pixels", with each pixel containing a dedicated ADC. The sensor contains three tiers – with the top tier consisting only of photodiodes, the second tier containing the ADC which generates spikes at a rate proportional to photocurrent, and the third tier contains an 8-bit counter to count the number of generated spikes. Since each pixel contains an ADC, this sensor offers opportunities for massive level parallelism by A/D converting all the pixels simultaneously. In addition, due to the absence of analog circuitry in the pixels, this architecture offers power scaling advantages, and allows trade-offs between performance and energy consumption.

Similarly, when it comes to the neural computation layer, we have two principal options. Both architectures are able to implement deep neural networks and perform feature extraction/classification, but vary widely in their approach.

1. The first neural accelerator, presented in CHAPTER 3, is based primarily on the Neurocube [27] architecture, and is a purely digital implementation. The processing engine essentially consists of 16 parallelly operating segments arranged in a 4×4 grid, with each segment processing a section of the image obtained from the sensor. There are two possible configurations for this sensor – one with on-chip stacked DRAM where

all DNN weights are stored on chip, and another configuration with synaptic weights stored in off-chip DRAM.

2. The second neural accelerator, presented in Section 4.2, is a processing-in-memory architecture based on ISAAC[33]. The main processing engine consists of analog computations carried out by crossbar arrays composed of ReRAM. Since the main computation engine is analog, this architecture requires the crossbar inputs to undergo D/A conversion before vector matrix multiplication, and for the crossbar outputs to undergo A/D conversion to interface with the rest of the digital blocks in the system. For this architecture, three possible configurations were introduced which optimize performance, power, and storage. For our analysis, we will focus only on the performance and storage optimized configurations since the performance and energy consumption for ISAAC-CE and ISAAC-PE are quite similar with only slight differences.

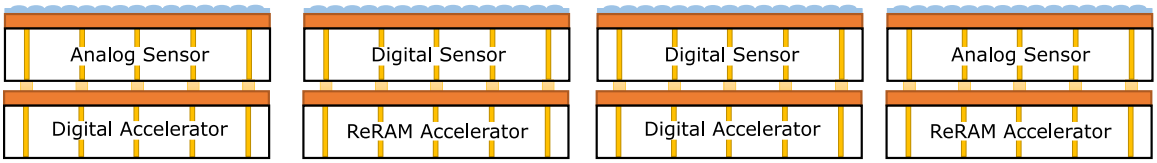


Figure 32 Overview of the four basic configurations of our system (off-chip DRAM not shown)

In order to investigate the impact of digital pixels and processing in memory on neural accelerators, and to compare against conventional sensing mechanisms coupled with digital architectures (i.e. Neurosensor), we are going to investigate four basic combinations of sensors and accelerators for our system, as shown in Figure 32. Also, since both the digital and ReRAM accelerators have multiple configurations (configurations C1 and C2

for digital accelerator, ISAAC-CE and ISAAC-SE for ReRAM accelerator), we have, in total, eight possible combinations of sensor and neural accelerator. It should be noted that all the possible configurations also contain off-chip DRAM, wherever applicable, to store synaptic weights in case the memory requirements are too large to fit on chip. In the subsequent section, we are going to discuss the simulation methodology to be used in order evaluate these different system configurations.

4.4 Simulation Framework

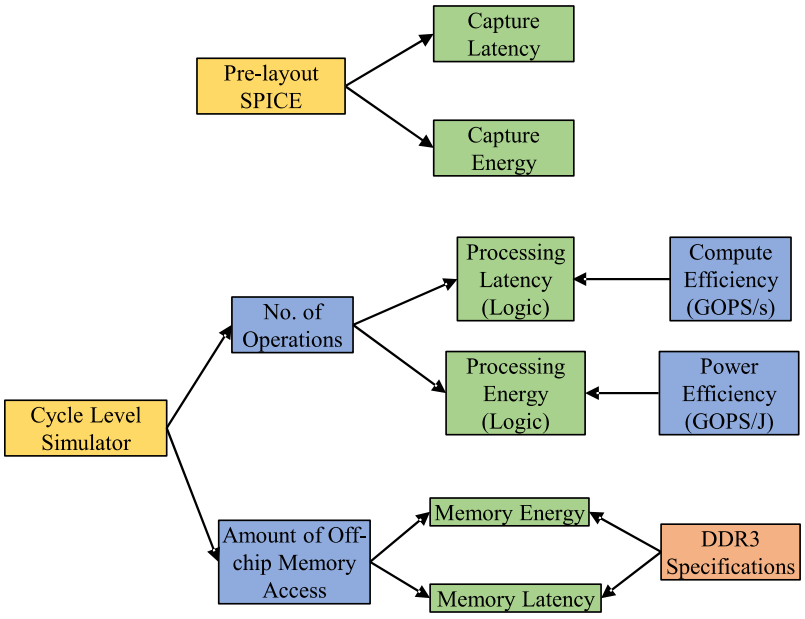


Figure 33 Latency and energy computation methodology for digital sensor and ReRAM accelerator

Our simulation framework analyses the system in terms of performance and energy. Similar to the analysis in CHAPTER 3, we consider only inference and assume the synaptic weights are loaded into the system from HPC at the very beginning.

For the analog sensor and digital neural accelerator, the same methodology as that in Figure 10 is used to determine computation latency and system energy. For the digital pixel

and ReRAM accelerator, we adopt the framework shown in Figure 33. Pre-layout SPICE simulations are used to determine the capture latency and energy. Since we do not have detailed access to the internal workings of the ReRAM accelerator, we adopt a simplified method to determine processing latency and energy. Using a cycle level simulator, we first obtain the number of floating point operations of a give neural network. Next, we divide the number of operations with the compute efficiency figure (in GOPS/s) quoted in [33] to find the processing latency. Similarly, for processing energy, we divide the number of operations (from cycle level simulator) with power efficiency (in GOPS/J) in order to obtain the processing energy. The cycle level simulator also states how much off-chip memory access, if any, is required. DDR3 specs are used to determine the resulting off-chip memory access latency and energy, similar to that in CHAPTER 3.

For the purpose of this analysis, we consider only computation, since both processing-in-memory and digital pixels affect only computation, and will have no effect on transmission latency or energy. The computation throughput, in terms of frames per second (FPS), therefore can be defined as

$$\text{Computation Throughput} = \frac{1}{t_{\text{capture}} + t_{\text{process}}} \tag{6}$$

where $t_{capture}$ is the time taken to capture the image, and $t_{process}$ is the time taken to perform feature extraction and/or classification on the captured image.

4.5 Simulation Results

4.5.1 ISAAC Performance, Energy Efficiency and Capacity

Table 8 System Parameters for ISAAC (PIM only, no sensor)

Parameter	ISAAC-CE	ISAAC-SE
IMA per tile	12	4
ADCs per IMA	8	8
Crossbar Arrays per IMA	8	512
Crossbar Size	128×128	256×256
Throughput (GOPs/s)	40902.33	8826.09
Power Efficiency (GOPS/J)	627.5	312.5
Storage Capacity (MB)	63.196	1729.35

As mentioned previously, we are going to evaluate two possible configurations for ISAAC, optimized for performance and storage; the power optimized configuration is not considered because its parameters, as well as performance, energy efficiency and storage are quite close to the performance optimized configuration. We calculate the performance, energy efficiency and storage efficiency of ISAAC considering the metrics reported in [33]. ISAAC-CE and ISAAC-SE have compute efficiencies of 478.95 GOPs/s/mm^2 and 103.35 GOPs/s/mm^2 respectively. Considering an area of 85.4mm^2, this puts ISAAC-CE throughput at 40902.33 GOPs/s and ISAAC-SE throughput at 8826.09 GOPs/s. However, this 4.5× reduction in throughput for ISAAC-SE is counterbalanced by a 27× increase in storage efficiency, which results in ISAAC-CE having a weight storage capacity of only 63.196 MB, while ISAAC-SE can store 1729.35 MB. This information is summarized in

Table 8. In the next section, we will use these throughput, power efficiency and capacity metrics to calculate the power and performance of our sensor integrated system according to the methodology in Figure 33.

4.5.2 Computation Throughput and Energy considering infinite storage

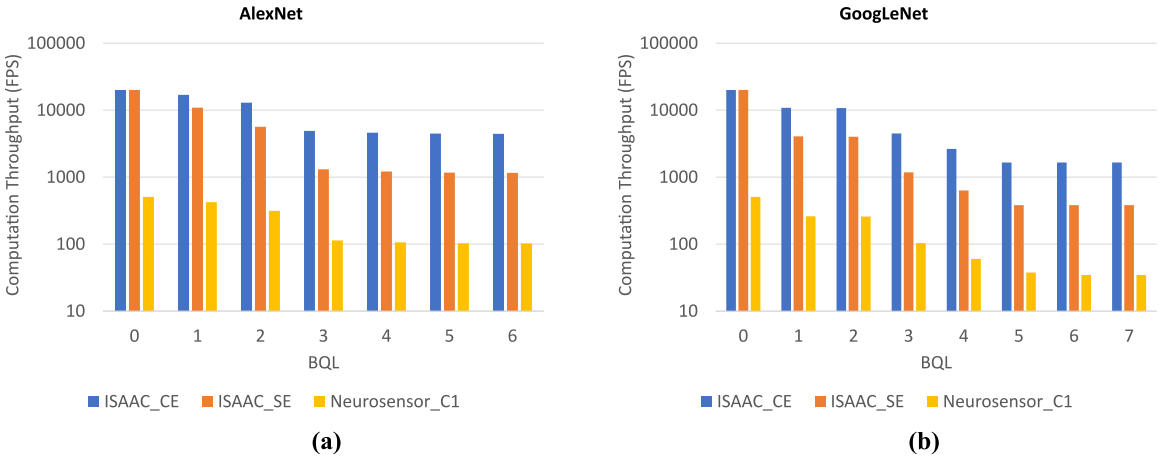


Figure 34 Computation throughput for (a) AlexNet and (b) GoogLeNet with integrated digital sensor. Memory limitations ignored. Neurosensor configuration C1 included for comparison.

For this analysis, we couple the PIM architectures with the digital sensor and investigate how the computation throughput and energy varies as more DNN layers are integrated in the system. Similar to the analysis in CHAPTER 3, we follow the same approach of block quantizing the convolutional neural networks. We are going to perform our analysis on two well-known convolutional neural networks – AlexNet, which has a relatively small number of operations (7.17 GOPs, 3661 MB memory) but large memory requirements, and GoogLeNet, which is heavy on computation but light on memory requirements (24.75 GOPs, 562 MB memory). For the initial part of our analysis, we are not going to consider the implications of limited on-chip weight storage capacity since our

objective is to initially demonstrate the full advantages of coupling digital sensors to PIM architectures without system level limitations imposed by area/memory capacity.

Figure 34 shows how the computation throughput for ISAAC integrated with digital sensor varies as more layers of the DNN are integrated on chip. Since the digital sensor offers an opportunity to scale performance, we have decided to operate the sensor with an effective capture time of 50µs, which translates to a frame capture throughput of 20000 frames/second. This represents a good compromise between sensor throughput and power, since running the sensor faster would increase power consumption, while running it slower would result in the sensor being the system bottleneck. As Figure 34 shows, integrating more layers on chip increases the amount of computation and thus decreases computation throughput, and ISAAC_SE has lower throughput than ISAAC_CE, as expected. The throughput for GoogLeNet is also lower than that for AlexNet due to the increased number of computations that GoogLeNet must carry out. When compared with configuration C1 for Neurosensor though, the computation throughput for ISAAC is significantly higher, even for the ISAAC_SE configuration, thus showing the possible advantages for processing-in-memory computation.

When it comes to computation energy, Figure 35 shows that PIM architectures also consume less energy than Neurosensor. ISAAC_SE exhibits inferior energy efficiency than ISAAC_CE since ISAAC_SE was primarily configured for increased storage capacity. Thus PIM architectures coupled with digital sensors exhibit high throughput as well as higher energy efficiency than Neurosensor.

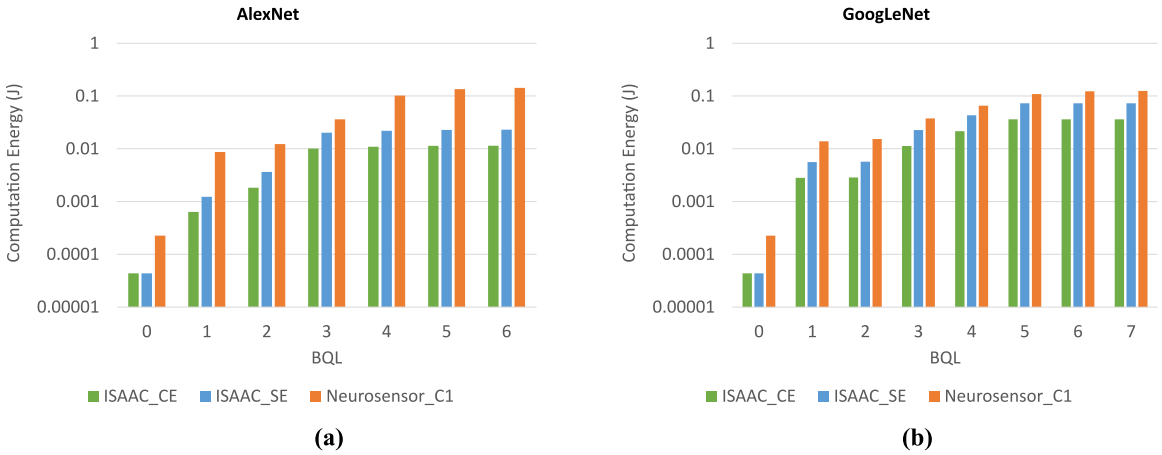


Figure 35 Computation energy for (a) AlexNet and (b) GoogLeNet with integrated digital sensor. Memory limitations ignored. Neurosensor configuration C1 included for comparison.

4.5.3 Computation Throughput and Energy considering limited storage

While the above analysis shows the performance and energy efficiency advantages of PIM architectures, a significant shortcoming of the above analysis is that we consider all the synaptic weights are stored on chip, and we do not consider the effect of having limited storage. In practice, the chip will have a physical storage capacity dictated by the design and the area, as shown in Table 8. Thus, only a fixed number of layers can be implemented on the chip at a given time. Once the layers have been fully processed, the weights for the next layer(s) have to be fetched from off-chip DRAM (similar to Neurosensor configuration C2, we assume a 32×16 = 512-bit bus for off-chip DRAM); this will introduce significant performance and energy overhead. Thus in this section, we are going to investigate how limited storage capacity changes our perception of PIM architectures.

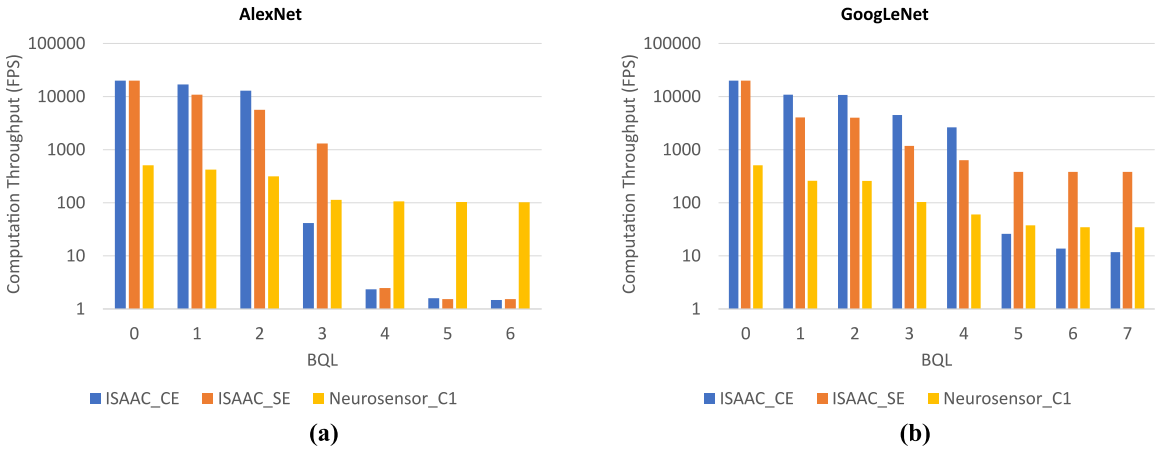


Figure 36 Computation throughput for (a) AlexNet and (b) GoogLeNet with integrated digital sensor considering limited synaptic weight storage. Neurosensor configuration C1 included for comparison.

Figure 36 shows the layer-by-layer computation throughput for AlexNet and GoogLeNet considering limited storage capacity for synaptic weights. As Table 8 shows, ISAAC_CE has a weight storage capacity of 63.196 MB while ISAAC_SE can store up to 1729.35 MB. Considering AlexNet, this allows ISAAC_CE to store only up to BQL 2, and for ISAAC_SE to store up to BQL 3. Whenever the storage capacity has been exhausted, the entire system must stall and wait for the weights to be fetched from off-chip DRAM. Similar to the analysis in CHAPTER 3, we assume the 512-bit DDR3 bus requires 12ns of latency for each fetch cycle. Because of this latency penalty for off-chip memory access, the throughput for ISAA_CE drops drastically at BQL 3; its capacity has been exhausted and new synaptic weights have to be fetched from off-chip memory. Similarly, after BQL 3, ISAAC_SE has to fetch weights from external memory and faces a corresponding drop in throughput. Therefore, even though the digital sensor architecture with ReRAM accelerator is superior to Neurosensor C1 for shallow networks or partial implementation of deep networks, its advantage is negated whenever off-chip memory access must be

performed. Neurosensor C1 is not subject to this memory access penalty because all the weights are stored in on-chip DRAM.

When GoogLeNet is considered in Figure 36(b), we see a similar story for ISAA_CE. The limited storage capacity only allows up to BQL 4 to be implemented on chip; subsequent layers require off-chip memory access and cause performance penalties. However, since GoogLeNet requires only 562 MB of memory, the network in its entirety can be implemented in ISAA_SE. As a result, it shows significantly higher computation throughput compared to Neurosensor C1 and ISAAC_CE.

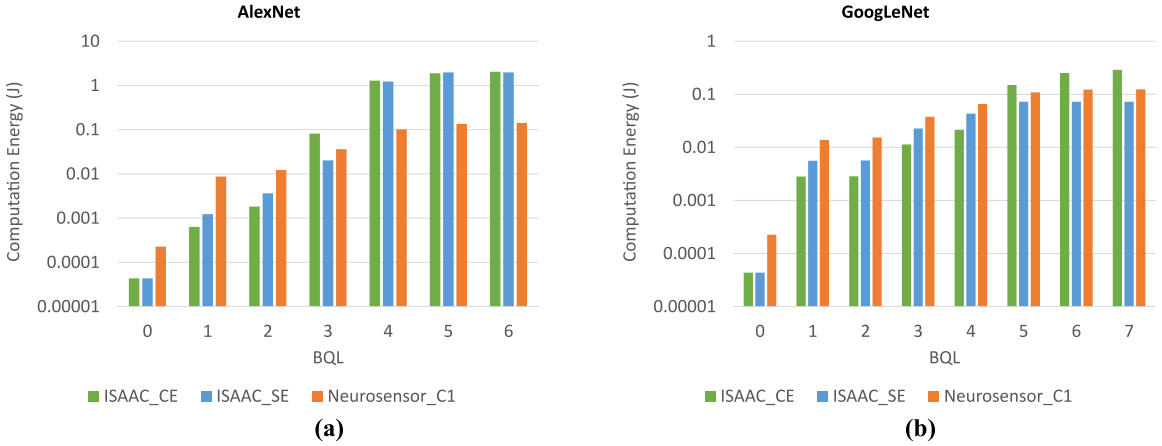


Figure 37 Computation energy for (a) AlexNet and (b) GoogLeNet with integrated digital sensor considering limited synaptic weight storage. Neurosensor configuration C1 included for comparison.

The energy consumption for PIM neural accelerators tells a similar story. Figure 37 shows that even though ISAAC_CE starts off as the configuration with the lowest energy consumption, as the network grows deeper, its energy efficiency advantages are cancelled out whenever off-chip synaptic weights must be fetched (BQL 3 for AlexNet, BQL 5 for GoogLeNet). ISAAC_SE fares better in this regard thanks to its higher storage capacity.

ISAAC_SE can thus maintain high energy efficiency up to BQL 3 for AlexNet and for the entirety of GoogLeNet.

To sum up our analysis for this section, it can be said that PIM architectures with integrated digital sensor retain their performance and energy advantages only as long as off-chip memory access is not required – this is expected, since if we want to access external memory, the architecture essentially no longer remains processing-in-memory. Thus for implementing deep neural networks with PIM architectures, we argue that rather than the computation or throughput efficiency, the capacity for synaptic weight storage is the key metric for determining performance and energy efficiency.

4.5.4 Impact of sensor architecture on throughput

So far we have seen how PIM architectures coupled with highly parallel digital sensors can be used to realize high throughput energy efficient neural accelerators (provided the memory requirements are kept low). In this section, we are going to investigate how the sensor architecture can determine the system throughput. To carry out this analysis, we will evaluate a number of different combinations by adopting a mix-and-match approach between sensor and accelerator architecture, as shown in Figure 32. Since both the ReRAM (ISAAC_CE, ISAAC_SE) and digital accelerator (configuration C1, configuration C2) have two variants each, we are going to evaluate a total of 8 different sensor-accelerator architectures.

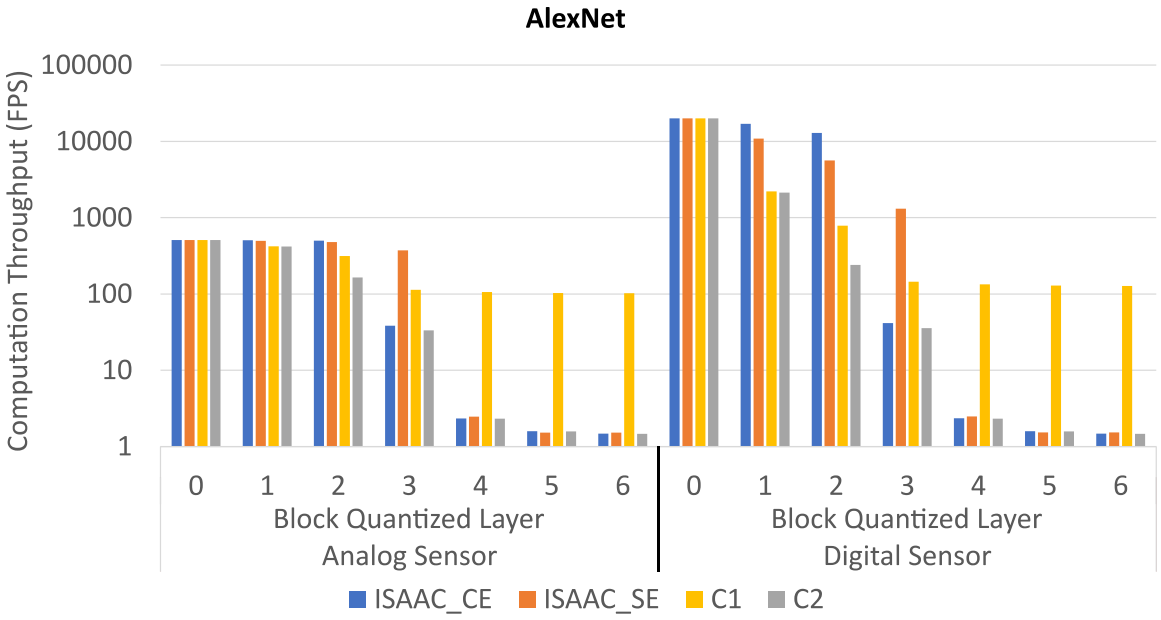


Figure 38 Impact of sensor architecture on computation throughput for AlexNet. ReRAM accelerator assumes limited storage for synaptic weights.

Figure 38 shows how the computation throughput for AlexNet changes as more layers are integrated in the system. We first couple the analog sensor architecture, presented in CHAPTER 3, with all the neural accelerators. It is seen that in the DNN BQLs for ISAAC where the throughput is not limited by memory capacity, e.g. BQL 1 and 2, the drop in throughput is almost negligible, which indicates that the processing latency (time taken to perform feature extraction) is negligible compared to the capture latency (time taken to capture and convert the image), thus image capture is the main performance bottleneck for ISAAC in this region. However, when ISAAC requires off-chip memory, e.g. BQL 4 onwards, there is a significant drop in throughput brought forward by off-chip memory access latency. The results for C1 and C2 coupled with analog sensor show steadily decreasing throughput with deeper networks from the very first layer, indicating that processing latency is the main bottleneck for the digital accelerators. It should be noted that C1 does not suffer the memory access penalty for deep networks since all its weights

are stored in on-chip DRAM, whereas for C2 which stores synaptic weights in off-chip DRAM, we see significant drops in throughput whenever a large amount of memory is being accessed (e.g. at BQL 4).

When it comes to the digital sensor, we initially see a very high throughput at BQL 0 (due to parallel sensing architecture). For the subsequent layers, both configurations (C1 and C2) for the digital accelerator show a large drop in throughput, indicating that the high throughput advantages for the digital sensor are being negated by the relatively long processing latency for both C1 and C2. When the entire DNN is implemented, the digital sensor offers only modest performance gains for the digital accelerator. The ReRAM accelerators, however, have much lower processing latency than either C1 or C2, and thus show much higher throughput than the case where they were bottlenecked due to the analog sensor. However, as discussed previously, this performance advantage is annulled whenever synaptic weights have to be fetched from external memory.

A similar analysis carried out using GoogLeNet echoes the previous trends to a large degree. However, since GoogLeNet has a lower memory requirement than AlexNet, it serves to better demonstrate the advantages of PIM computation since it allows more layers to be implemented on chip before being subjected to off-chip memory access penalty.

Figure 39 shows that for analog sensor, even though ISAAC_SE is initially bottlenecked by long capture time, the reduced processing time enabled by PIM architecture causes it to achieve the highest throughput among the four evaluated combinations. ISAAC_CE also initially exhibits high throughput, however it is subjected to a throughput drop whenever off-chip memory access is required (BQL 5 onwards). C1

and C2, on the other hand, are not bottlenecked by the sensor, and shows a steady reduction in throughput as the network gets deeper and more processing is performed. Similar to the AlexNet analysis, digital sensors for C1 and C2 do not provide large advantages since they are primarily limited by the long processing time. However ISAAC shows significantly heightened throughput as long as off-chip memory access is not required.

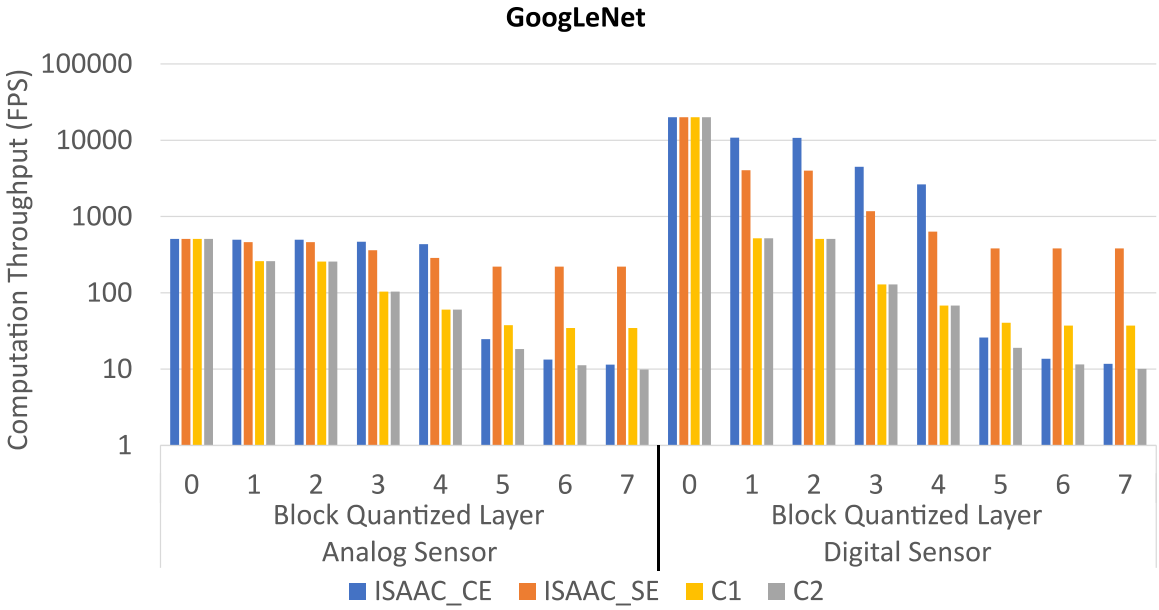


Figure 39 Impact of sensor architecture on computation throughput for GoogLeNet. ReRAM accelerator assumes limited storage for synaptic weights.

When it comes to energy, Figure 15 and Figure 37 show that capture energy is a negligible component of the total computation energy. Thus any impact of the sensor architecture on the total computation energy will be minimal; therefore we do not investigate how different sensor architectures will impact the energy efficiency of the system.

To summarize this discussion on how processing-in-memory computing and digital sensors can achieve higher energy efficiency, Figure 40 shows the computation energy and

throughput for all the eight configurations when the entire GoogLeNet classification pipeline is implemented. Firstly, it is seen that off-chip memory access always entails a large drop in throughput and corresponding increase in energy – hence, C2 as well as ISAAC_CE exhibits reduced energy efficiency due to their dependence on off-chip memory. Also, since these configurations are bottlenecked by memory access latency, the impact of sensor architecture is negligible. C1 exhibits superior energy efficiency due to its on-chip DRAM, however, the digital accelerator is still not fast enough to take full advantage of the digital sensor, and ends up bottlenecking it, thus making both configurations for C1 fairly identical. ISAAC_SE, on the other hand, is the only accelerator fast enough to fully utilize the throughput advantages of the digital sensor, and the combination of ISAAC_SE with the digital sensor offers the highest throughput as well as lowest energy efficiency; all the other configurations end up bottlenecking either the sensor or the accelerator.

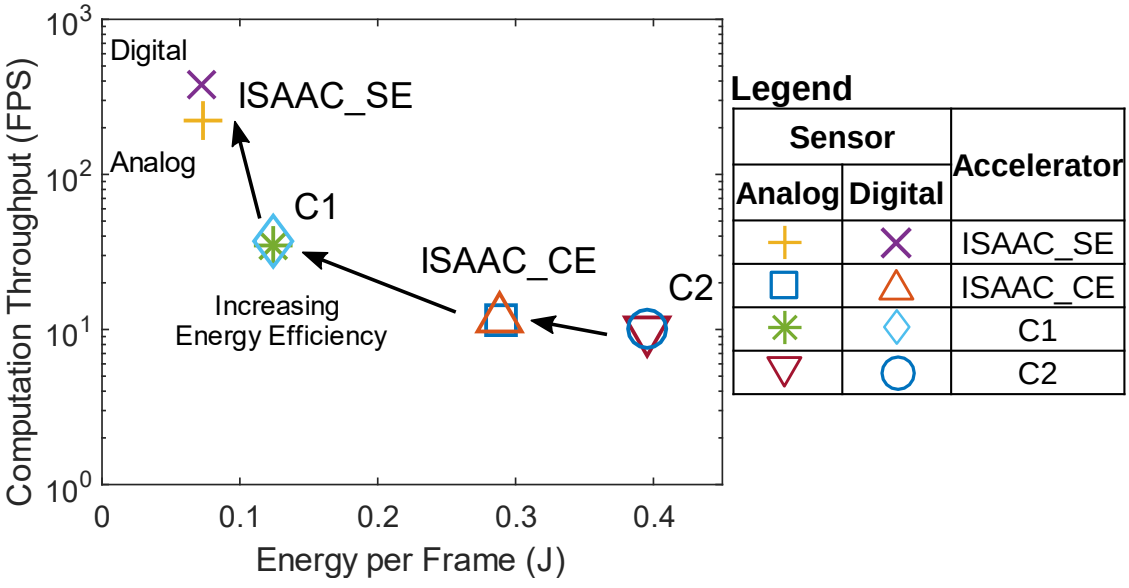


Figure 40 Throughput vs energy for varying accelerator and sensor architectures considering GoogLeNet classification

4.6 Summary

This chapter started with the Neurosensor concept put forward in CHAPTER 3, and explored alternate options for the sensor architecture and neural accelerator. While Neurosensor contained primarily an analog sensor coupled with a digital accelerator, this chapter investigated the implications of coupling a digital sensor with a primarily analog accelerator based on ReRAM. The digital sensor offers opportunities for achieving extremely high throughput imaging by integrating an ADC for each pixel and A/D converting all pixels simultaneously. The ReRAM accelerator also offers opportunities to achieve increased throughput due to its implementation of processing-in-memory computing. However, the potential throughput advantages are largely dependent on the amount of memory required by the implemented neural network – and off-chip memory access in order to access synaptic weights often negates the advantages of PIM computation. Subsequent analysis was also carried out to investigate how sensor architecture affects the throughput. Our analysis found that coupling the digital accelerator with the high throughput digital sensor provides only modest improvements in performance due to processing latency being the main bottleneck for the digital accelerator. Similarly, coupling analog sensor with the ReRAM accelerator diminishes the performance advantages of PIM computing since it is bottlenecked by the capture latency. However, as the network grows deeper and computation latency becomes more dominant, the PIM architecture throughput advantages become more apparent despite the slow sensor architecture. Thus our analysis found that the configuration with the highest energy efficiency involves coupling the high speed analog accelerator with the high speed digital sensor; other configurations end up bottlenecking either the sensor or the accelerator. In

summary, massively parallel digital sensors stacked with PIM architecture based neural accelerators can further leverage the advantages of 3D integration and help achieve high performance and energy efficient sensing and classification platforms. However, the system (accelerator) configuration, in particular the memory capacity, and the architecture of the of DNN itself remains one of the key determinants of performance and energy efficiency gains.

CHAPTER 5. RECONFIGURABLE IMAGE SENSOR NODE WITH ENERGY HARVESTING

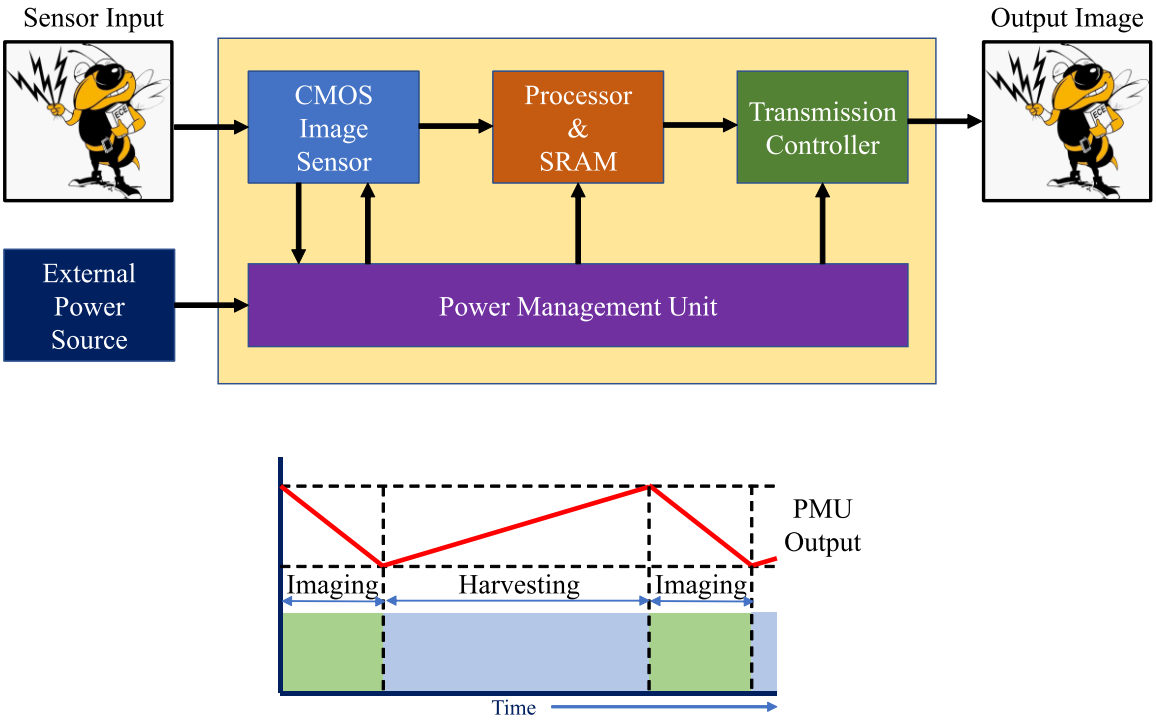


Figure 41 System overview of image sensor node

This chapter discusses the design and post-silicon measurement results of a 2D image sensor node with integrated energy harvesting capabilities [46, 76]. Figure 41 shows the component blocks of the chip, designed in 130nm technology. The image sensor array and ADC capture and convert the image in 8×8 pixel macroblocks which facilitates block level pipelining. The SRAM block acts as a buffer to the processor. The digital processor determines whether a macroblock contains moving objects (Region-of-Interest, ROI) or not. Only the ROI MBs are transmitted, while non-ROI MBs are dropped, thus leading to transmission energy savings. The wireless transmission controller acts as an interface to an off-chip transmitter.

The system is powered by an energy storage element (battery or high-density capacitor), which can be charged by the energy harvested from the image sensor. The image sensor can be configured to work in either imaging or harvesting mode. Under imaging mode, the sensor works in photoconductive mode, performing image capture and A/D conversion, and subsequently passing the converted image to the processor. All the system blocks are powered by the power management unit (PMU) under this mode. However, under harvesting mode, the power management unit turns off power to all non-essential blocks, and the sensor is configured to operate in photovoltaic mode, essentially turning into a solar cell and allowing energy to be harvested by the power management unit and stored in an off-chip capacitor. The PMU output voltage thus increases during harvesting mode when energy is being stored in the capacitor, and decreases during imaging mode when energy is being drawn from the capacitor. The ultimate goal of this work is therefore to achieve self-sustained operation, where the system is able to operate solely with energy harvested from the sensor.

It should be mentioned that the development of this chip was a collaborative effort, with the author of this work being involved primarily with the design of the sensing and power management blocks. Details of the ROI-based compression mechanism can be found in [64], and is not a focus of this thesis. Therefore, for the purpose of this work, we will assume our chip to be a system which generates energy in harvesting mode, and performs image capture and transmits the unprocessed/uncompressed image through an off-chip wireless transmitter in imaging mode. Hence we will constrain our discussion only to the sensor, ADC and the power management circuits. Initially, we will discuss the design

and findings from the first version of this chip, and subsequently discuss the modifications and results from a revised version of the same system that was taped out at a later date.

5.1 System Overview

5.1.1 Energy Harvesting Image Sensor

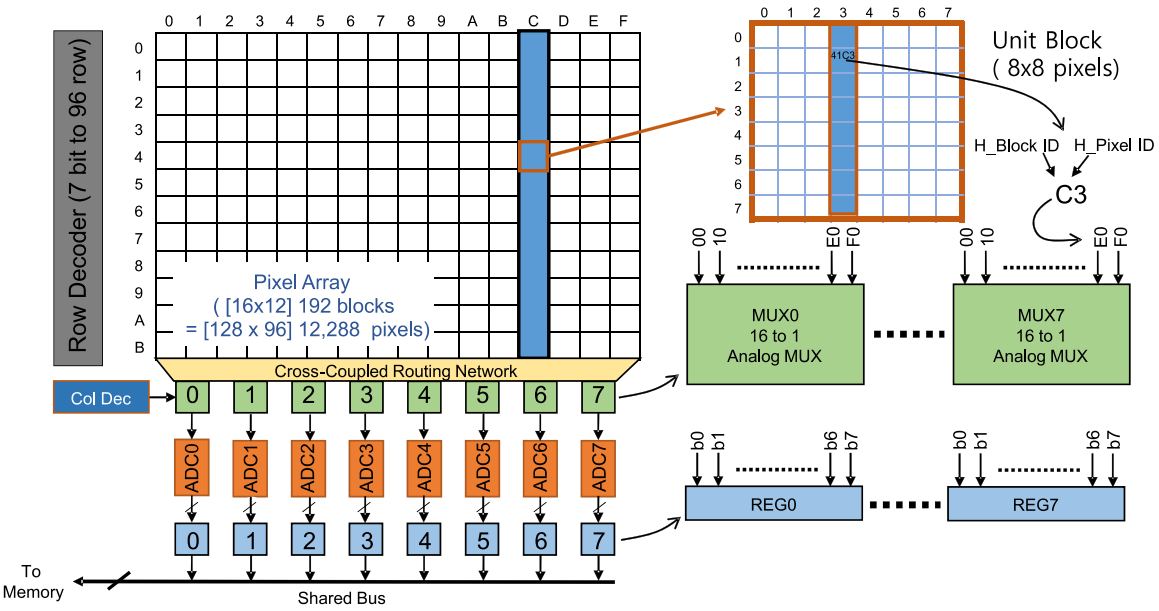


Figure 42 Block diagram of CMOS image sensor

Figure 42 shows the system schematic of the image sensor array, designed in 130nm. The CMOS image sensor array contains 128×96 pixels arranged into 16 horizontal and 12 vertical blocks, each of which consists of 8×8 pixels. Rather than the conventional method of row-wise readout, the CMOS sensor performs a block-wise readout which enables block level processing and pipelining between the sensor and the processor. For block level readout, after the first row is selected, the first 8 columns are read out simultaneously through 8 parallel ADCs. Once conversion is finished, the ADC outputs are latched. Rather than the conventional approach of reading out the next 8 columns of the first row, for block-level readout, we select the next row and the first 8 columns of the second row are read

out. This continues until the 8th row of the block is read out, which completes the read out of the first block. We then move on to the next block and the entire readout procedure is carried out again, 8 columns at a time, until all the 128 macroblocks are read.

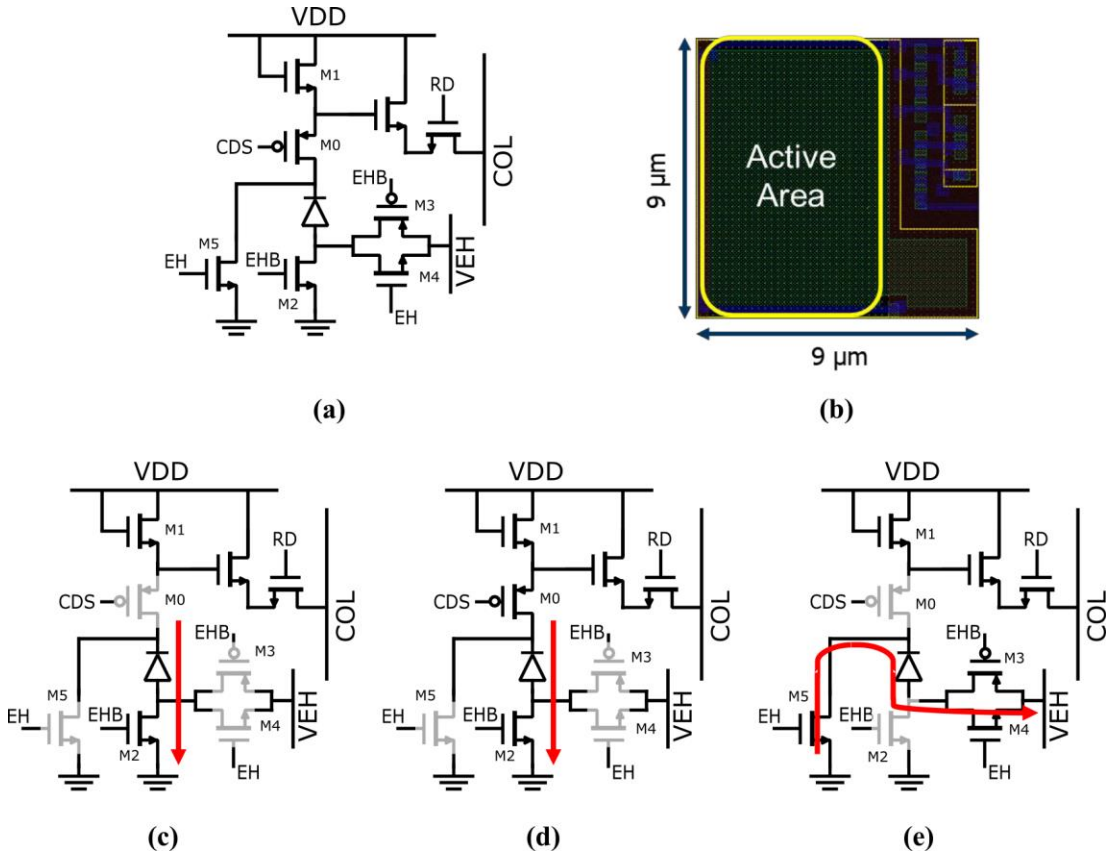


Figure 43 (a) Circuit schematic of logarithmic energy harvesting pixel (b) Pixel layout (c) Imaging mode operation, CDS dark sample (d) Imaging mode operation, CDS illuminated mode (e) Harvesting mode operation

Figure 43 (a) and (b) show the circuit schematic and layout respectively of the logarithmic energy harvesting pixel. Compared to the standard logarithmic pixel seen previously in Figure 2, this pixel contains four extra transistors which enable it to be configured for either imaging or harvesting mode. The size of the pixel still remains the

same as presented previously at 81μm^2, however the addition of four extra harvesting transistors causes a ~10% drop in fill factor to 44%.

Figure 43 (c), (d) and (e) explain how the pixel operates in imaging and harvesting mode. Under imaging mode, the harvesting signal EH is low, which essentially reduces the pixel circuit to the standard logarithmic pixel of Figure 2. This pixel also employs CDS in order to minimize the effect of fixed pattern noise, the operation of which can be seen in Figure 43 (b) and (c). During harvesting mode, the pixel is configured as seen in Figure 43 (e). This turns off the signal readout path, and connects the photodiode cathode to the ground, and energy is extracted at the V_{EH} node. The V_{EH} node of all the pixels are shorted together to effectively form a single photovoltaic cell. In this mode, the charge flow from the pixels will be similar to that of a photovoltaic cell with an effective area of 0.44mm^2 (128×96 pixels × 81μm^2 × 44% fill factor).

The analog-to-digital converters are 8-bit single slope ADCs, and adopt an architecture co-designed for correlated double sampling. The ADC architecture is identical to that in 3.1.2., and A/D converts the difference between the dark and the illuminated samples. There are 8 parallelly operating ADCs, and the sensor output voltages are muxed into the input of these 8 ADCs. Once A/D conversion is completed, the digital outputs are transferred to the image processor through a 64-bit (eight 8-bit ADCs) bus.

5.1.2 Power Management Unit

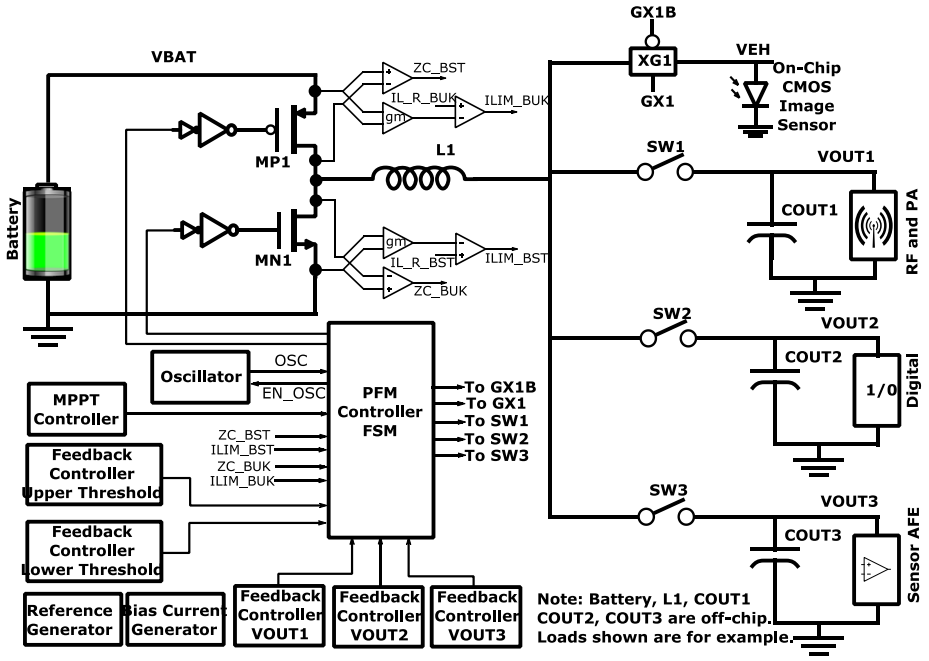


Figure 44 PMU architecture with energy harvesting and voltage regulation [76, 77]

Figure 44 shows the energy harvesting and power delivery system architecture which uses a single inductor and single power stage [76, 77]. The PMU architecture in Figure 44, employed in the first version of the chip, was also a collaborative effort, and further details of the architecture can be found in [77]. The contribution of this thesis towards the power management unit is the identification of the limitations of this architecture, and implementing modifications to improve system functionality. The PMU modifications have been discussed in Section 5.3.4, and have been implemented in a revised version of this testchip.

The power management circuit operates in two modes – harvest (boost operation), and power delivery (buck operation). The pulse frequency mode (PFM) controller provides all the control signals for the boost and buck operation. During harvesting mode, the transmission gate XG1 is turned on, which transfers energy through boost converter

operation from the photodiode (VEH) to the storage element (VBAT). During power delivery mode, the PMU transfers power from the storage element (VBAT) to VOUT1, VOUT2, and VOUT3 through buck converter operation. During this mode, the transmission gate XG1 is turned off, and the switches SW1, SW2, and SW3 are turned on using a dedicated feedback controller for each output. The output voltage rails have built-in priority control to provide opportunity to differentiate between critical and non-critical circuit blocks (VOUT1 has the highest priority while VOUT3 has the lowest). Cross-regulation is provided by this priority based mechanism, where, in the event of simultaneous power demand, power is provided to the lower priority voltage rail only when the demand for the higher priority output has been met. This ensures no cross regulation at the highest priority output while offering standard cross regulation performance at the lowest priority one.

The control circuits are powered by VBAT allowing harvesting from very low voltages when VBAT>0.37V. The PMU also contains zero-current detection and over current protection for both buck and boost mode operation. The maximum power point tracking (MPPT) controller uses fractional MPPT technique [78, 79], and limits the inductor current if the loaded VEH drops below 50% of the open circuit VEH. This MPPT technique thus simplifies the algorithm while harvesting 70% of the peak power. Autonomous bias gating cuts off bias current to the non-essential system blocks in sleep phase for both buck and boost modes, thus causing significant reduction in bias power [80, 81].

The PMU also contains autonomous mode management which controls switching between the buck and boost mode of operation depending on priority of load and battery

voltage level. For applications with a target frame rate, a frame rate control signal will cause the harvesting signal, EH to change from low (sensing) to high (harvesting). When EH is low, the PMU is totally in power delivery mode, and only acts as a SIMO buck converter to provide power to the sensing, processing and transmission blocks. When EH is high, it enables both buck and boost operation. The boost mode harvests energy from the sensor to the battery. However, the buck operation is essential for some circuit blocks even between two frames. For example, preserving the frame data for the previous frame is essential when performing frame differencing based compression, and hence the memory must always be powered on. The autonomous mode ensures (i) boost-only operation when battery voltage (VBAT) is less than a specified lower-limit threshold (ii) buck-only operation when battery voltage (VBAT) is higher than a higher-limit threshold, and (iii) switching between buck and boost mode operation when VBAT is between the higher-limit and lower-limit thresholds.

5.2 Measurement Results

In this section we are going to discuss the results from post-silicon characterization of the image sensor. A test-chip (shown in Figure 45) in 130nm GF8RF process was fabricated to demonstrate the image sensor operation. The dimensions of the chip are 2mm×2mm, and the chip is wire bonded in open cavity LCC68 package. The inductor of the power management unit is off-chip, and integrated into the PCB.

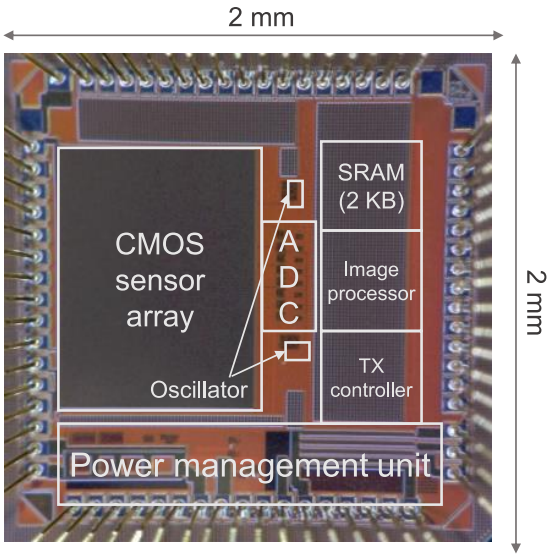


Figure 45 Die photo of the image sensor chip

5.2.1 Image Sensor and Energy Harvesting

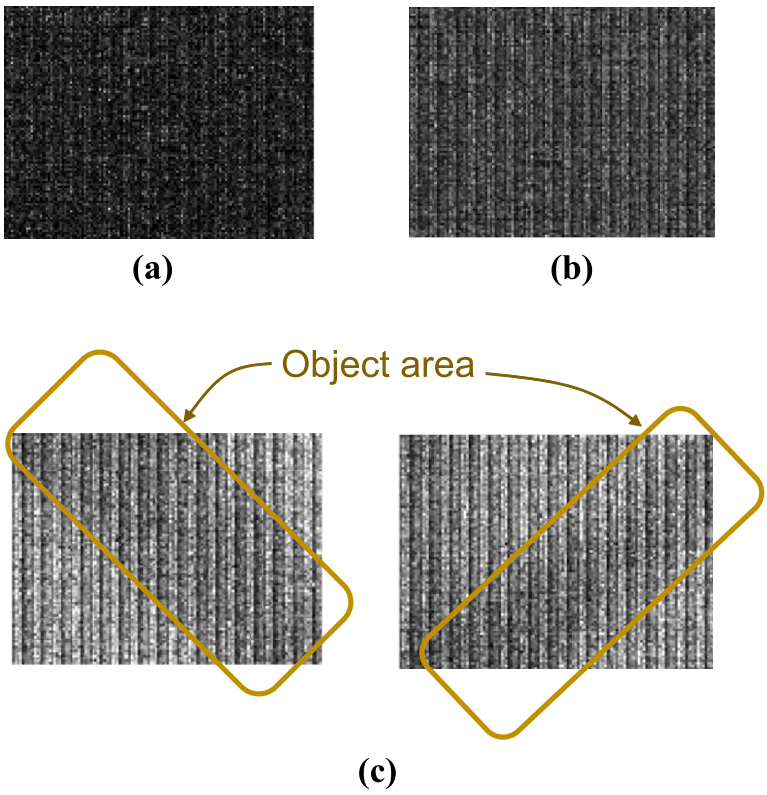


Figure 46 Sensor output under (a) fully dark condition (b) 180klux illumination. (c) Image captured with a thin object in front of sensor

Functionality of the image sensor was tested by interfacing the chip with MATLAB. The transmission controller in the chip is used to send out a stream of serialized data which is captured by a data acquisition unit (NI PXIe-1082). The data acquisition unit interfaces with MATLAB, which is then used to reconstruct the image from serialized data. Figure 46 shows images captured for fully dark and illuminated conditions as well as with a static thin object in front of the sensor in different positions. The image shows high level of random and fixed-pattern noise. Also, the pixel value differences between the object (dark) and the background (bright) are not significant, indicating very limited dynamic range (5.42dB) and reduced pixel photosensitivity. All these factors contribute to an image that unfortunately provides very limited information. In a later section, we will discuss the contributing factors behind the poor imager performance and the steps we can take to improve it.

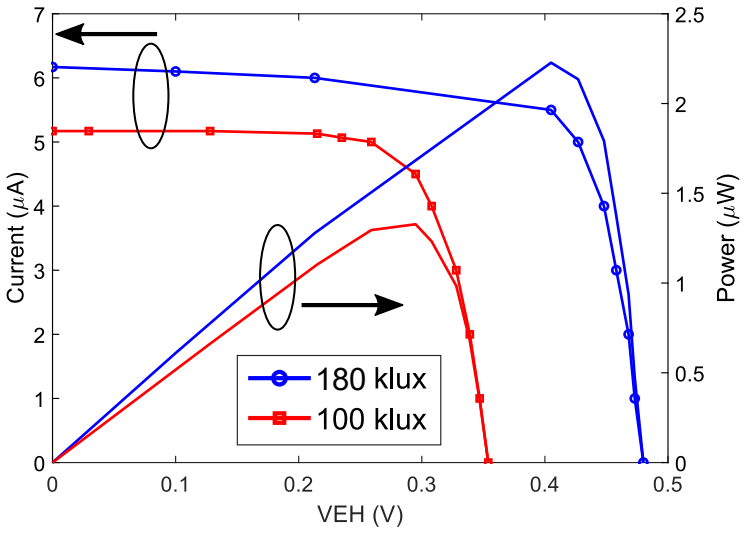


Figure 47 I-V characteristics of the sensor energy harvesting under 100klux and 180klux intensity

Figure 47 shows the DC I-V profile of the CMOS pixel array during harvesting. A cool white (7000K) LED lamp was used as a light source for harvesting. The peak power

was measured to be 2.1µW at 180klux luminance.

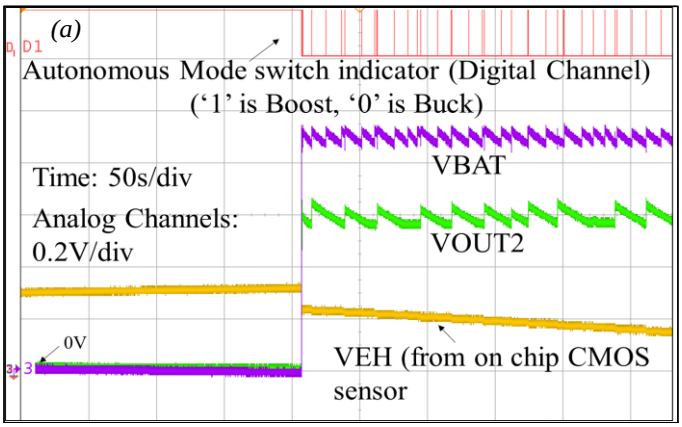


Figure 48 PMU operating with energy harvested from CMOS image sensor [77]

Figure 48 show the PMU harvesting from the image sensor, storing energy in battery and supplying a load domain. It is seen that VEH keeps decreasing as power is being drawn from the storage capacitor. Figure 49 show how the efficiency of the buck and boost regulators change with load. The efficiencies are low primarily because of unoptimized switch sizing.

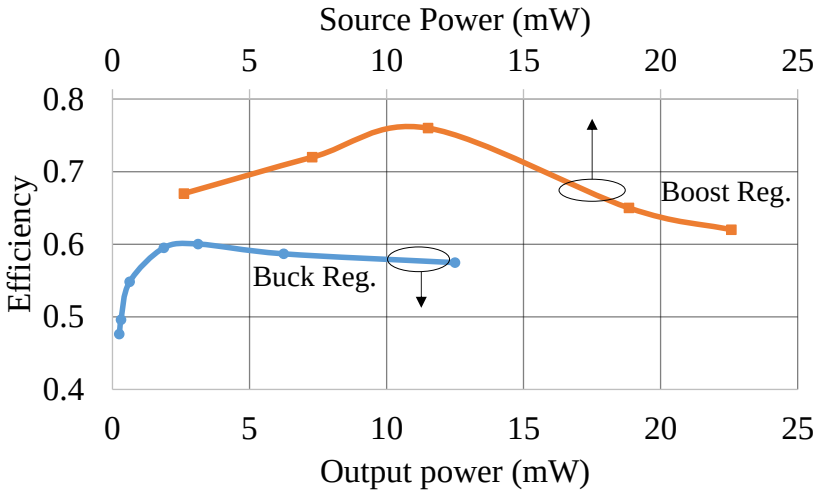


Figure 49 Efficiency profile of the boost and buck regulator [77]

5.2.2 System Self-Powering

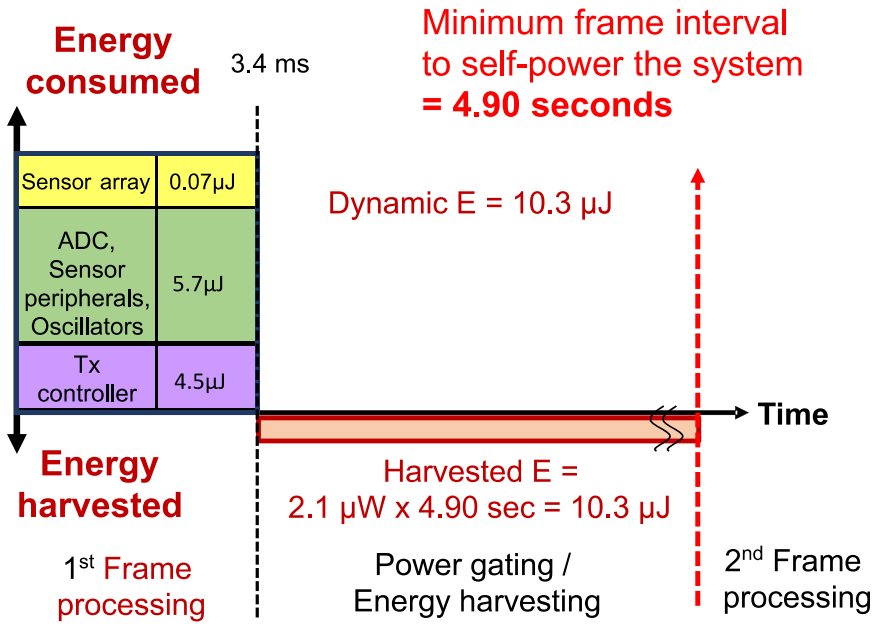


Figure 50 Breakdown of capture energy and break-even point for self-sustained operation (excluding transmitter)

Based on measured power generation and consumption of the sensor, we estimate the self-supported frame rate (frame/sec). Figure 50 shows the consumed/harvested energy over time, neglecting the transmitter energy and latency overhead. At the maximum operating frequency of the system, processing of one frame (image sensing, processing, and preparing packets for transmission) consumes 10.3µJ of dynamic energy. After transmitting the last block of the frame, the system switches into the harvesting mode. During the harvesting mode, the entire system components are power-gated to avoid the leakage energy consumption. The image sensor array in the harvesting mode generates 2.1µW assuming 180 klux light intensity. Therefore, this power should be integrated over time to supply the dynamic energy of the system for frame processing (10.3µJ). In this

setup, assuming energy can be extracted at 100% efficiency at the maximum power point from the sensor, the minimum frame interval for self-powered operation is 4.90 seconds.

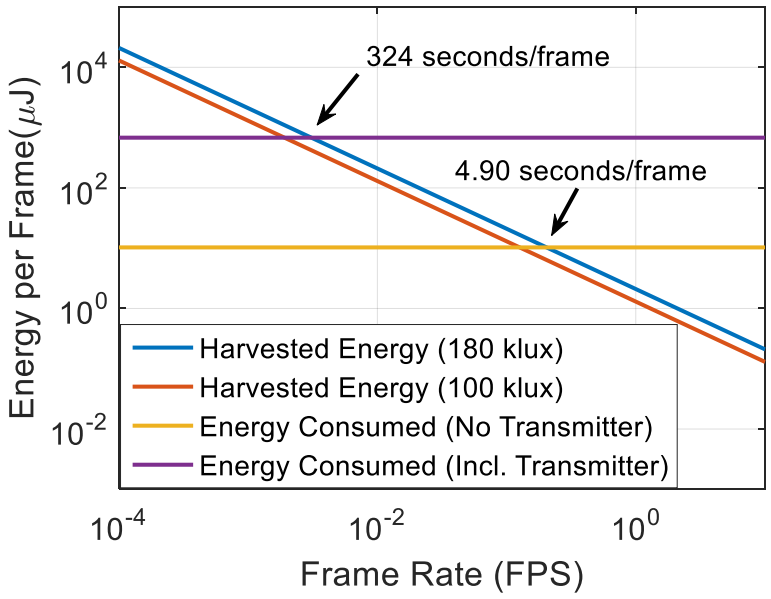


Figure 51 Harvested/consumed energy for varying frame rate and illumination

To project the self-powered performance under varying frame rate, we show in Figure 51 how the frame rate changes when the harvested energy powers the on-chip system. In addition to the scenario in Figure 50, we investigate harvesting under 100klux illumination, and also explore how the self-powered frame rate changes if a transmitter is integrated into the system. It is seen that as frame rate decreases, harvested energy increases because of increased harvesting time. As seen from Figure 50, to obtain self-sustained operation, the interval between each frame capture event must be greater than 4.9 seconds. If we wish to also power the transmitter with the harvested energy, we must also account for the transmission energy overhead. Using a low power transmitter such as the nRF52840 [82] consumes 500µJ to transmit 98.3kbits (single frame) of data. In addition, we must also account for the additional time the system needs to stay awake in order to interface with

and send the image into the transmitter, which increases the frame processing time to 60ms and consumes further energy since the system needs to stay longer in imaging mode. Thus considering the transmitter adds both latency and energy overheads to our system. Assuming these overheads, the self-powered frame capture interval increases to 324 seconds/frame. Considering reduced light intensity (100 klux) further increases the self-powered frame intervals to 7.9 s (without transmitter) and 524 s (with transmitter).

Table 9 Key parameters of the image sensor chip (first iteration)

Specification		**Without Transmitter**	**With Transmitter**
Maximum Power @ 180 klux (μW)		2.1	2.1
Maximum Power @ 100 klux (μW)		1.3	1.3
Energy Consumption per frame (μJ)	Sensor Array	0.07	1.24
	ADC, peripherals	5.7	101
	TX Controller	4.5	79.4
	Transmitter	N/A	500
	Total	10.3	681
Minimum Frame Capture Time (ms)		3.4	60
Self-powered frame interval @ 180 klux (s)		4.9	324
Self-powered frame interval @ 100 klux (s)		7.9	524

So far we have discussed the design and measurement results of our image sensor node with energy harvesting, and the principal measurement results are provided in Table 9. We have also identified several limitations of the design. The image sensor exhibits poor photosensitivity and dynamic range. In addition, the output of the sensor is too noisy to

provide useful information. On the harvesting and power management side, we have seen that poor photosensitivity causes the generated power to be low, and coupled with the low PMU efficiency in the low power region, this makes the system unable to sustain self-powered operation. In the next section, we will investigate the reasons behind the poor performance, and consider possible modifications to improve the overall design.

5.3 Design Modifications for Performance Improvement

5.3.1 Imager Noise Improvement

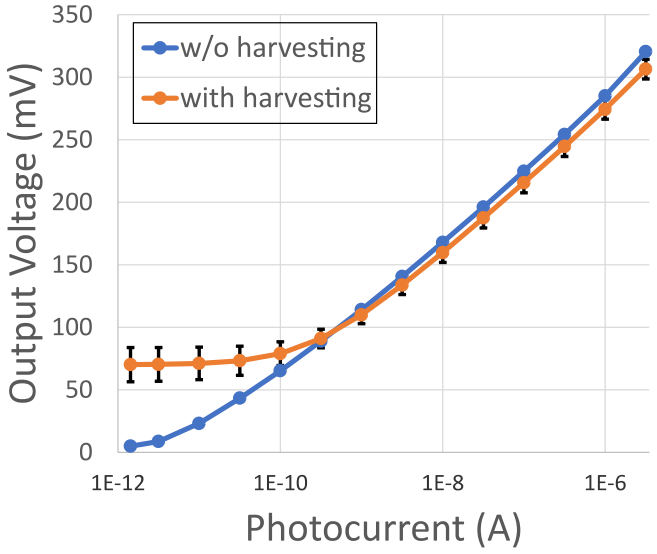


Figure 52 Pixel response of harvesting pixel and non-harvesting pixel. Error bars represent standard deviation of photocurrent response

In order to examine the sensor noise, we performed noise analysis on the harvesting pixel similar to the methodology laid out in Section 3.4.3. A pixel photocurrent sweep was carried out to find the pixel response to different levels of illumination, and 1000-point Monte Carlo analysis was performed at every point in order to model the change in pixel output due to transistor variation. Figure 52 shows the pixel response of the harvesting

pixel (Figure 43) considering FPN and compares it against the standard logarithmic pixel (Figure 2). The harvesting pixel exhibits higher FPN, especially at low photocurrent levels; in addition the harvesting pixel also has lower dynamic range.

The discrepancy in photocurrent response between the two pixels can actually be traced back to the harvesting transistor M5. Under normal imaging conditions, we originally neglected the leakage current through M5. However, once we consider leakage current through transistor M5, equation (2) actually becomes

$$V_{ACT} = V_{DD} - V_{th,M1} - nV_T \ln\left(\frac{I_{ph} + I_{0,M5}}{I_0}\right) \tag{7}$$

where $I_{0,M5}$ is the subthreshold leakage through M5. Under this condition, (3) becomes

$$\Delta V = V_{ACT} - V_{RES} = nV_T \ln\left(\frac{I_{ph} + I_{0,M5}}{I_{leak,M0}}\right) \tag{8}$$

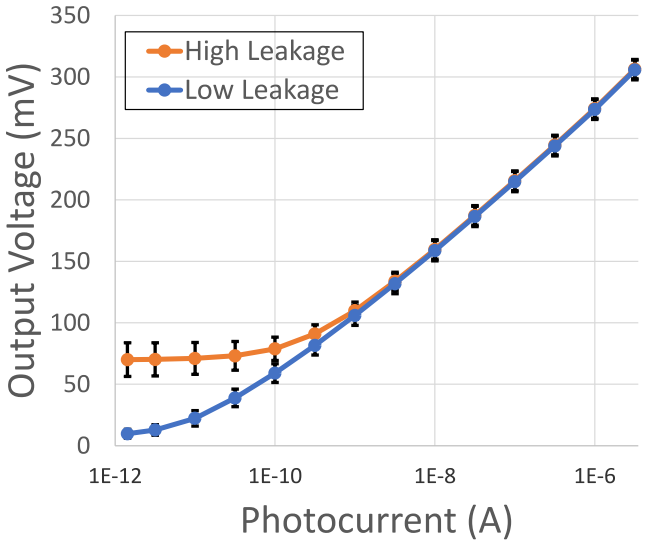
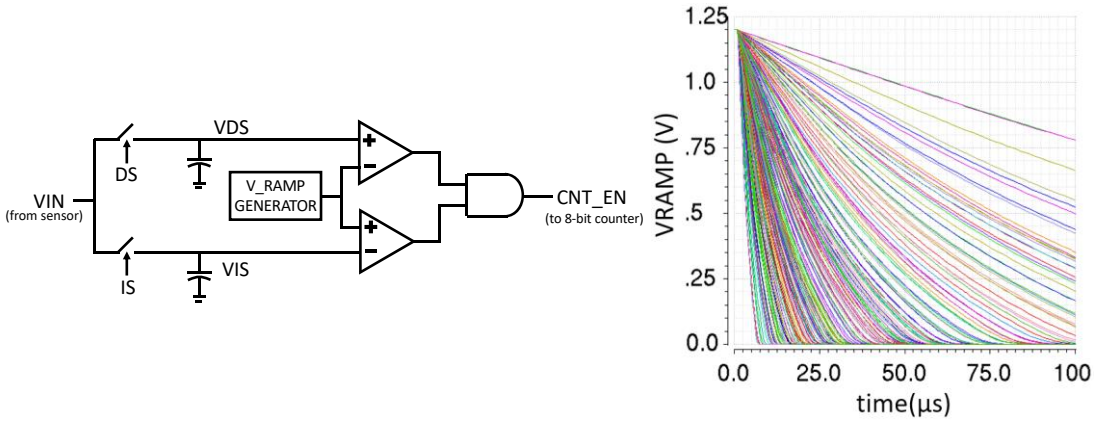


Figure 53 Pixel output for harvesting pixel with high leakage and low leakage harvesting transistor. Error bars show standard deviation of photocurrent response

Thus it is seen that despite CDS, there will be FPN due to subthreshold leakage current through the harvesting transistor M5. In addition, at low photocurrents, when the photocurrent becomes comparable with leakage current through M5 (~100pA from simulation), the pixel becomes unresponsive to light, and thus exhibits poor dynamic range. Since the main reason behind the poor dynamic range and high noise is leakage current through M5, replacing M5 with a high-threshold, low leakage device should serve to mitigate noise as well as restore dynamic range, as seen in Figure 53.

5.3.2 ADC Noise Improvement



(a) **(b)**

Figure 54 (a) ADC architecture (b) Variation in ramp voltage waveform for 100 Monte Carlo runs

The above analysis points out one of the reasons behind the noisy image in Figure 46. However, if the only source of noise is the image sensor pixel, the noise should be uncorrelated. Contrary to that assertion, Figure 46 shows a clear noise correlation among pixels in the same column, with vertical lines of noise running down the image. Therefore, it also becomes necessary to examine the ADC readout mechanism for possible sources of noise. For the ADC, we use the same architecture as seen in Figure 3. The principal sources of noise in the ADC are the sampling switches, sampling capacitor leakage, comparator

offset, and ramp voltage generator. In the current design, each ADC contains its own ramp generator, which causes different discharge rates for the ramp waveform among the 8 ADCs. In addition, the ramp capacitor is typically discharged with a low current (~10nA), which further deteriorates the situation and increases variation in the ramp rate from one ADC to the next. Figure 54(b) shows the ramp waveform for 100 Monte Carlo runs; this large variation in ramp rate is primarily responsible for the vertical line FPN in the sensor image, as it causes a marked difference in the A/D converted values among neighboring pixels. If, however, we use a single ramp generator for all ADCs, as shown in Figure 55, and the same reference waveform is passed to all the 8 ADCs, this problem will be eliminated, and the biggest component of vertical FPN will be mitigated. In addition, using a single ramp generator will allow us to use a bigger ramp capacitor and discharge it with a larger current, thus further reducing detrimental effects due to variation.

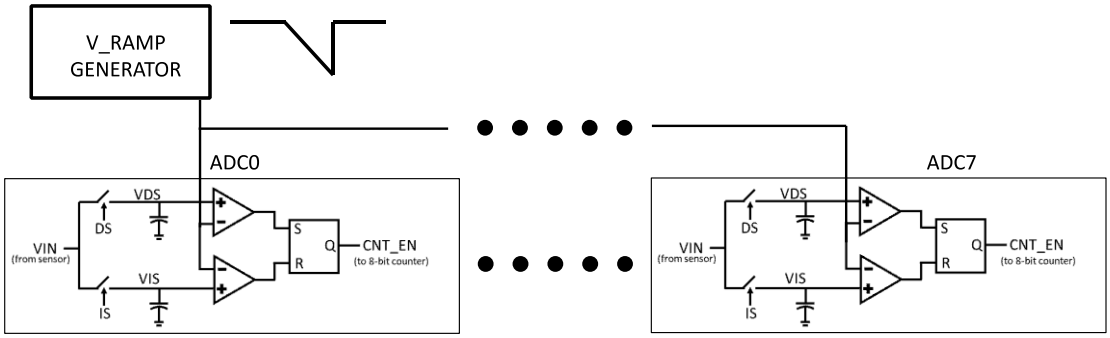


Figure 55 Modified ADC architecture - central ramp generator for all ADCs

In addition to the major architectural change outline above, some minor modifications were also added to the ADC architecture to improve performance and reliability. Previously, the 8-bit ADC counter did not have overflow protection. So it was quite possible, if the ramp rate was too slow, for the ADC count to overflow, thus reporting erroneous values. In addition, another shortcoming of the ADC architecture was that its

operation was entirely self-timed. Capture and conversion of the next cycle would start immediately after all the 8 ADC were finished with A/D conversion. This would lead to an image dependent variable frame rate since darker images (less conversion time due to a lower) would be converted faster than well-illuminated images. The ADC design was thus modified to include counter overflow protection, and its architecture was modified so that the conversion time (frame rate) was dependent solely on the clock frequency.

5.3.3 Harvesting Power and Photosensitivity Improvement

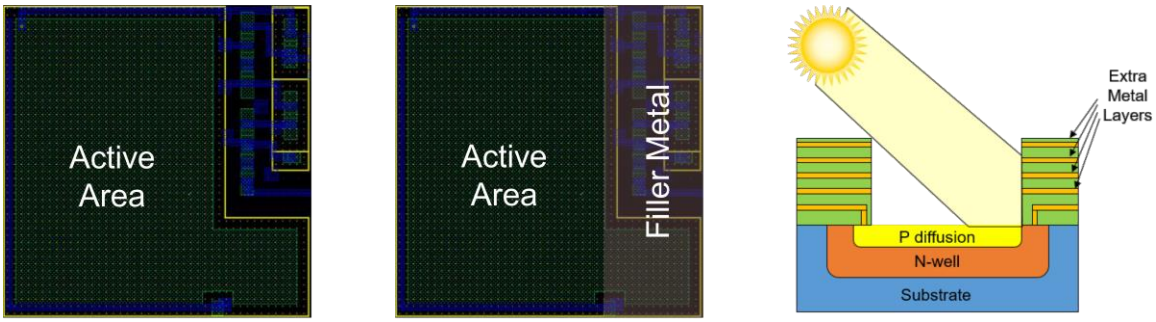


Figure 56 (a) Pixel layout without filler metal (b) Pixel layout with filler metal (c) Cross-section of pixel with filler metal

One of the major factors towards limiting self-powered frame rate is the harvested power from the sensor itself. A major contributor towards this limitation comes from the extra metal layers in the pixels that must be placed to satisfy the minimum metal density rule enforced by the foundry. As Figure 56 shows, placing the extra metal layers adjacent to the pixel places the photosensitive area in a "well". For the 130nm technology that we use for fabrication, the depth of this well is 21.4µm [83], which significantly impedes the flow of light into the sensor.

A possible method to mitigate this effect is to increase the area of the unit pixel by

4× to 18µm × 18µm (Figure 57). Using a bigger pixel will reduce the amount of filler metal (by percentage) in the pixel, and will thus allow more light to fall on the photosensitive area. In addition, increasing the pixel area also increases the fill factor from 44% to 70%, which should lead to a further increase in the amount of power generated. A bigger pixel should also increase photosensitivity due to increased photodiode area. It should be noted that since we want to keep the total chip area (2mm × 2mm) unchanged, an increased pixel size will result in a decrease in resolution from 128×96 pixels to 64×48 pixels. Lower image resolution results in lower perceptual quality of the image to the user, however this does not significantly degrade the detection performance of the moving object detection method.

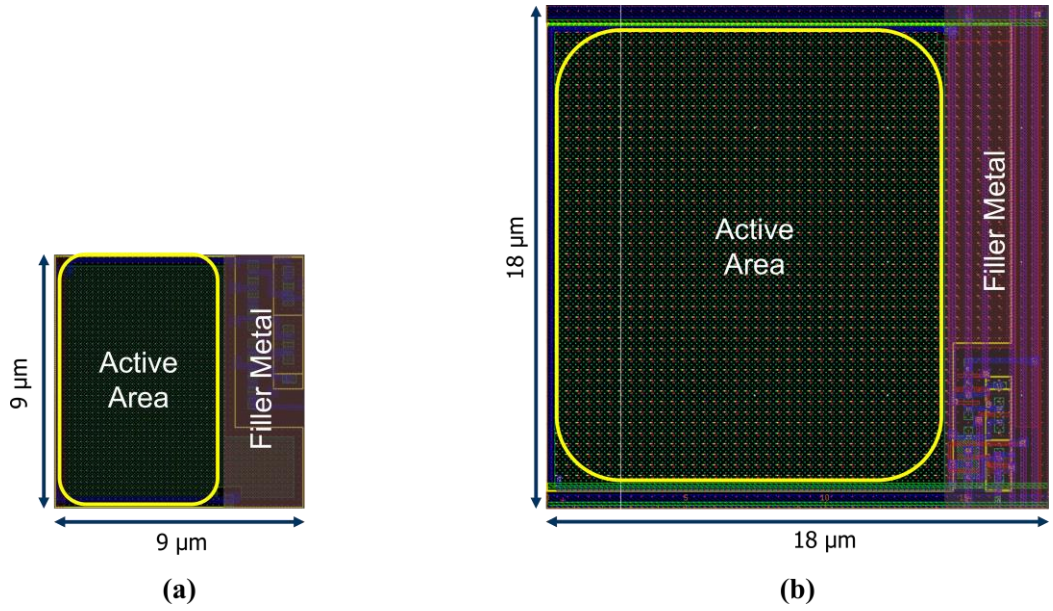


Figure 57 (a) Original pixel layout (b) Modified pixel layout with increased area

5.3.4 Power Management Unit Improvement

One of the primary hurdles towards self-powered operation is the low harvesting efficiency of the power converter. Since the PMU exhibits low efficiency in the µW range,

it cannot effectively harvest energy into the storage capacitor. The main reason behind this comes from the SIMO single power stage architecture of the system. During imaging, the system requires power in the mW range, which requires large drivers and power FETs, however, harvesting power is in the neighbourhood of µW. Using the same large drivers and power FETs for this range causes excessive switching loss, and thus reduces efficiency.

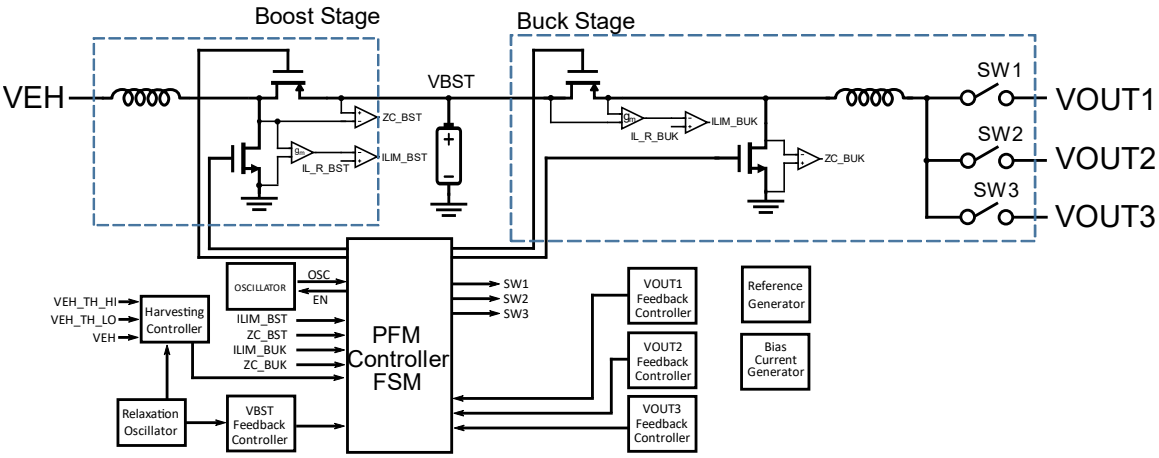


Figure 58 Modified PMU architecture with dedicated power stages for buck and boost

Figure 58 shows the modified PMU architecture with dedicated power stages for buck (harvesting power) and boost (delivering power). Since the power stages have been split up, it is now possible to size them independently for the power ranges they will be operating at. Since the boost stage is still operating at the mW range, the PFET and NFET power FET sizes are kept unmodified at 20mm and 10mm respectively. For the boost stage, the lower power requirements mean that the PFET and NFET power FETs can be downsized to 8mm and 4mm respectively. This downsizing also leads to reduction in the size of the power FET gate drivers, causing further savings in power. In the next sections, we discuss further modifications of the buck and boost architectures.

5.3.4.1 Boost Architecture Modifications

The following major changes were made to the boost converter in order to improve its efficiency and minimize power consumption.

1. More aggressive bias gating of oscillator, zero-current detection, and current limit blocks. The previous iteration of the architecture did not bias gate these circuit blocks when idle.
2. Redesign of reference generation block to reduce power consumption. Resizing the bandgap generator circuit caused the bias current consumption of the bias generator to drop from 25nA to 11nA.
3. Replacement of the MPPT block by a hysteretic comparator with externally adjustable harvesting thresholds. Excessive current being drawn from the harvesting capacitor might cause the voltage at node VEH to fall to extremely low levels at which boosting is not possible. The comparator monitors the voltage at the VEH node and stops harvesting whenever the voltage drops below the lower harvesting threshold. The boost converter starts up again when the voltage at VEH node reaches the upper harvesting threshold.

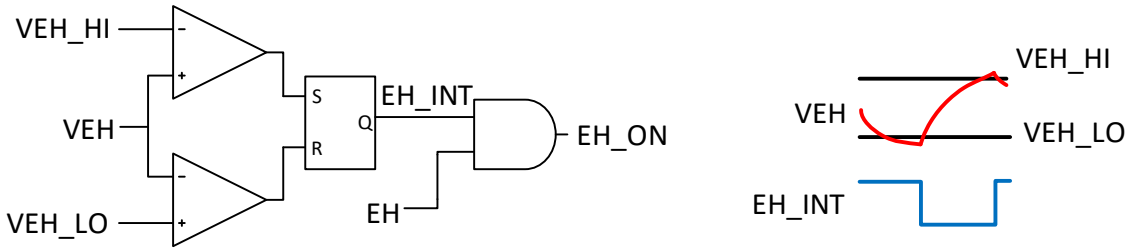


Figure 59 Threshold based harvesting controller

4. Use of a relaxation oscillator [84] to minimize comparator bias current. In the previous iteration, the feedback comparators in the boost architecture always stayed on, thus continually consuming bias power. A relaxation oscillator, which is essentially a low

power, low frequency oscillator with 33% duty cycle, can be used to bias gate the oscillator, and thus reduce the bias power consumption.

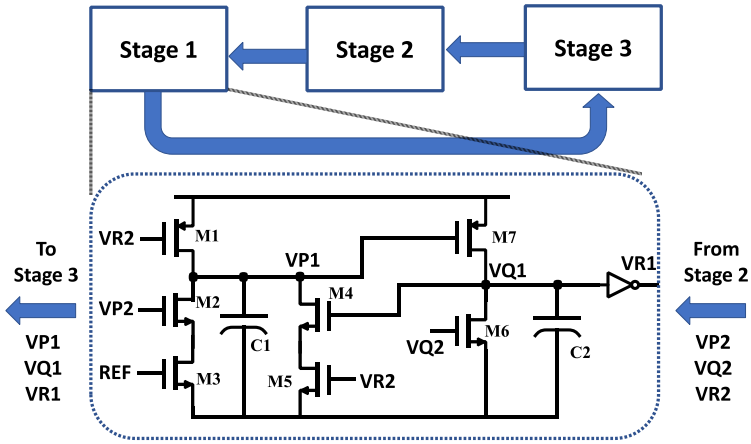
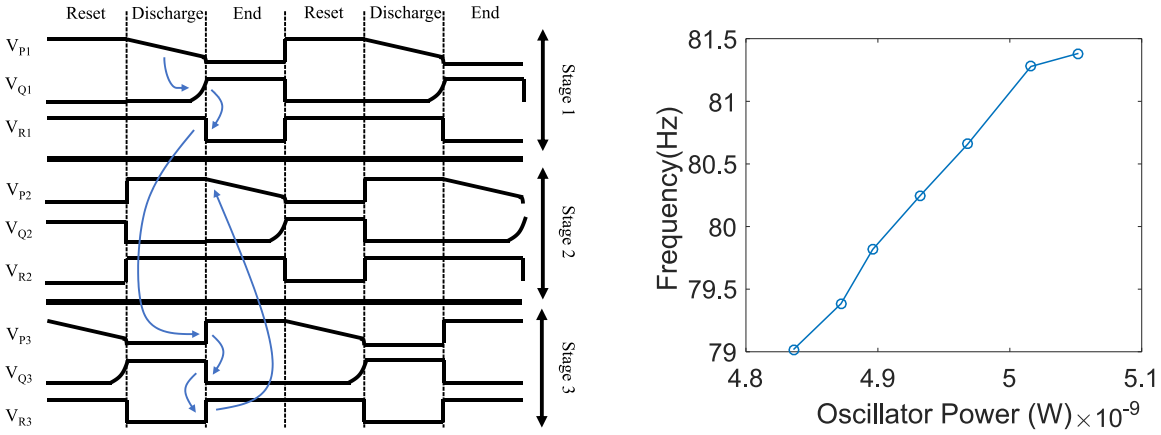


Figure 60 Relaxation oscillator architecture

Figure 60 shows the architecture of the relaxation oscillator. The core of the oscillator consists of three delay stages arranged in series, and the timing operation is shown in Figure 61(a). Each stage generates a delay proportional to the input current (set by the signal REF). Starting from Stage 1 at the reset stage, node VP1 is pulled up to VDD by M1 and VQ1 is pulled down to 0 by M6. At the timing stage, M2 turns on, which discharges the capacitor at a rate proportional to the input current. When VP1 reaches Vtrip, M7 turns on and VQ1 is pulled up to VDD by M7 and positive feedback through M4. The next timing cycle is started by the node VR1, which causes stage 3 to go into reset mode; the whole cycle then repeats for stage 3 and then stage 2. Figure 61(b) shows how the oscillator frequency scales with power.

Thus, to summarize the major changes to the boost converter, a hysteretic threshold based harvesting controller has been implemented, the bias generator has been resized, the oscillator and current limit comparator have undergone more aggressive bias gating, the

power FET drivers have been reduced, and a relaxation oscillator has been integrated to periodically bias gate the feedback comparators. The savings in idle power (from simulation) due to the above changes can be seen in Table 10.



(a) **(b)**

Figure 61 (a) Timing waveform for relaxation oscillator (b) Power versus frequency of relaxation oscillator

Table 10 Idle current consumption of boost converter (simulation results)

System Block	Previous Design (nA)	Revised Design (nA)
Bias Generator	25.6	11.2
Oscillator & Current Limit Comparator	19.6	0.43
Comparators	20	9
Power FET Drivers	5	2
Relaxation Oscillator	N/A	4.5
Total	70	27

5.3.4.2 Buck Architecture Modifications

Aside from the splitting up of the power stages, the architecture of the buck converter

remains largely unchanged. Changes have been made to how the voltage rails behave in response to the harvesting signal. The first voltage rail, VOUT1, is meant to supply the sensor and preprocessor logic which can safely be turned off during harvesting. Therefore, VOUT1 is configured such that the harvesting signal causes SW1 in Figure 58 to turn off, thus turning off power to VOUT1. The third voltage rail, VOUT3, is used to power the circuit blocks that remain constantly on, such as the frame rate controller and harvesting signal generator; hence the switch SW3 which powers VOUT3 is independent of the harvesting signal. Finally, the analysis in [46] has shown that the self-powered frame rate can be considerably increased if the SRAM supply voltage is scaled during harvesting mode. Therefore, the feedback controller for the second supply rail, VOUT2, contains two independently adjustable voltage regulation levels. During image capture and processing, VOUT2 will be regulated at the higher voltage level, and during harvesting, VOUT2 will be regulated at the lower voltage level, thus resulting in power savings and increasing the self-powered frame rate. Table 11 shows the output specifications.

Table 11 Buck converter output specifications

Voltage Rail	Priority	Imaging Mode	Harvesting Mode
VOUT1	1	ON	OFF
VOUT2	2	ON, regulate at high voltage	ON, regulate at low voltage
VOUT3	3	ON	ON

5.4 Measurement results for the revised test chip

The revisions mentioned in the previous section were implemented and a revised version of the testchip was taped out again (Figure 62). The dimensions (2mm × 2mm) and package (LCC68) remain unchanged from before. Due to a separate power stage for the boost converter, this chip requires an extra inductor compared to previous iteration, which is integrated off-chip in the PCB. In this section we are going to go over the preliminary test results from the revised chip.

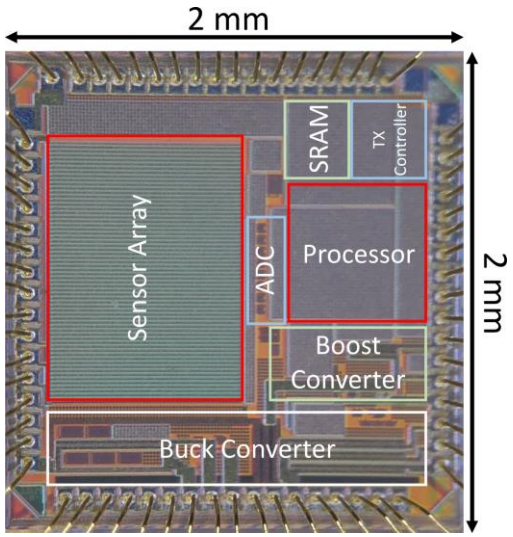
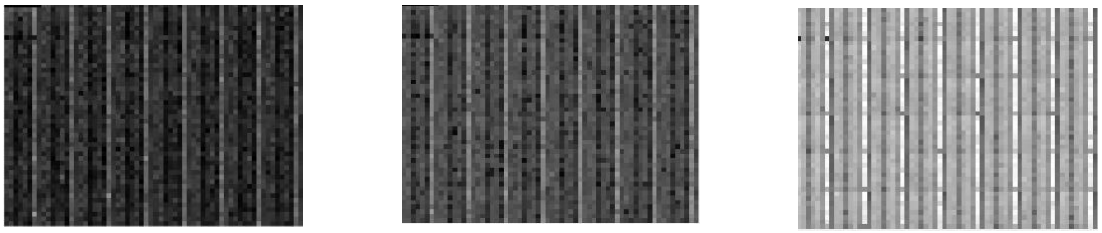


Figure 62 Die photo of the revised chip

5.4.1 Imager Performance Results

Figure 63 shows the sensor output under three different illumination conditions. Compared to the output of the previous chip in Figure 46, we see considerably improved sensitivity to light owing to the changes implemented in the imager (bigger pixels, low threshold harvesting transistors) and the ADC (central ramp generator, counter overflow protection). In particular, the sensor now shows sensitivity to ambient light as well, which was not at all observed with the previous design, and exhibits superior dynamic range (10.67dB). It should be noted that in order to keep the sensor area unchanged while

increasing the pixel size, the sensor resolution had to be decreased to 64×48 pixels (from 128×96). Hence the size of the images in Figure 63 are actually 4× smaller than that in Figure 46.



(a) **(b)** **(c)**

Figure 63 Sensor output from the revised chip under uniform illumination (a) Completely dark condition (b) Under ambient light (c) Under 180klux illumination

It should be noted that even though redesigning the ADC has produced improved results, the images still show vertical lines of fixed pattern noise. In fact, there is one particular ADC (out of 8) which is primarily responsible for this noise, with a response noticeably brighter than its neighbors. This is mainly because the column line connected to the impacted ADC actually contains a test structure designed to feed external voltages for ADC testing. Thus this causes a variability in the ADC response due to slightly different loading on the column line. In order to improve this noise behavior, we post-process the image through a rudimentary interpolation method, which serves to mitigate this issue to a large degree, as seen in Figure 64.

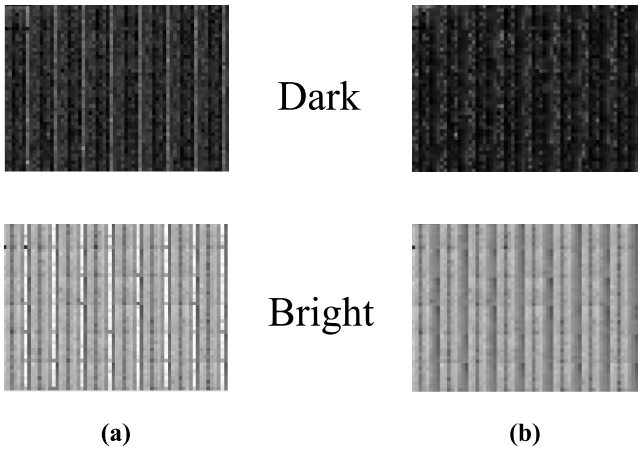


Figure 64 Sensor output (a) before interpolation (b) after interpolation

Next, we attempt to use the sensor to capture actual images. In order to do so, we require the sensor to be coupled with a lens which will focus an image of the object onto the sensor. In order to determine the proper focal length, we first use a F/2.0 aperture fisheye lens DSL218A-NIR-F2.0 [85] and couple it with an Arducam 2MP module [86]. We then calibrate the lens for proper focal length, and transfer the lens over to our sensor module.

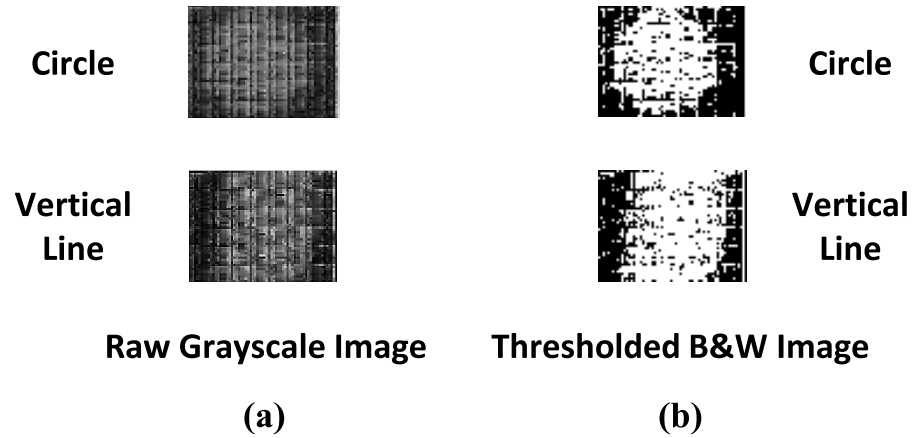


Figure 65 (a) Raw grayscale sensor output for circle and vertical line image pattern (b) Post-processed black and white image after thresholding

Figure 65 shows the sensor outputs for a light circle and a vertical region of brightness, captured with the fish-eye lens. While the noise performance and image quality are better than before, the performance is still not optimal. However, Figure 65(b) shows that threshold-based post processing can be used to produce black and white images where the patterns are more visible. It should be mentioned that we have also not been so far able to produce more intricate patterns using the sensor, and the challenges related to the sensor testing are discussed below.

5.4.1.1 Imager Performance Testing Challenges

One of the principal challenges to testing the sensor comes from the optical issues. As explained previously, we used a fish-eye lens to determine the focal length. However, lenses are typically meant to be used with a specific sensor size. Improper matching of sensor and lens would cause optical issues such as vignetting or image cropping.

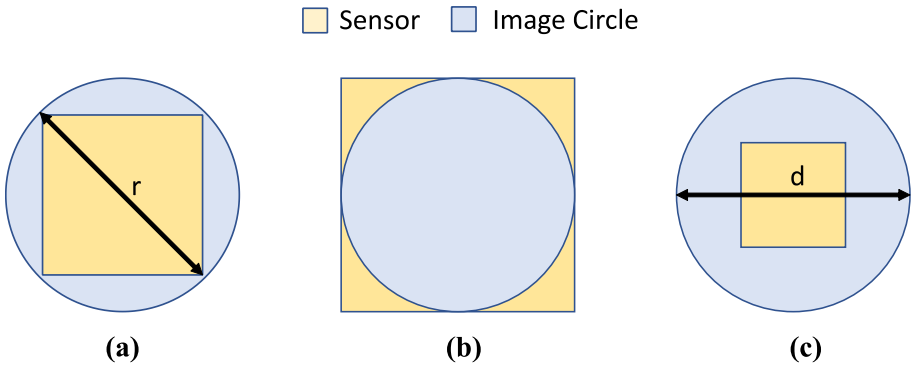


Figure 66 (a) Proper matching of lens and sensor (b) Vignetting - sensor bigger than the lens image circle (c) Image cropping - image circle too big for sensor

As Figure 66 shows, for proper matching, the sensor diagonal r and the lens image circle d must be equal. However, if the sensor is much bigger than the image circle, there would be parts of the sensor outside the image circle on which no light would fall, and the

sensor output would produce images darkened at the edges. This phenomenon is called vignetting, and is shown in Figure 66(b). On the other hand, if the image circle is much bigger than the sensor, the sensor would only capture part of the image, essentially cropping the rest of the image out, as shown in Figure 66(c).

For our sensor, the diagonal length was 1.44mm, while the lens with the smallest possible image circle that we could find was 3mm. Figure 67 shows how this crop factor would affect the captured image when moving from the 2MP Arducam sensor, which had a diagonal length of 4.48mm, to our sensor with a diagonal length of 1.44mm. As can be seen from Figure 67(b), the small sensor size causes a severe reduction in the viewing angle. Thus the object (the lamp with the pattern) must be placed very accurately in front of the lens due to the narrow viewing angle, making alignment difficult. While we have tried placing the object further away from the lens in order to facilitate alignment, it was found that moving the object further away reduces the intensity of light falling on the sensor, thus resulting in the sensor not being able to properly respond to low intensity light.

(a) **(b)**

Figure 67 (a) Image captured with 2MP Arducam (b) Image cropping due to small sensor size

Figure 68 Light bloom overpowers fine details

One of the reasons why we have so far been unable to proceed with more complicated patterns relates to the light bloom phenomenon, as seen in Figure 68. Essentially, when there is a high intensity light source behind an object with fine details, the light source shines through and diffuses around the object as if it was not there at all. Therefore when we tried to make more complicated patterns on our light source, the light diffused around the patterns, and an almost uniform brightness image was captured by our sensor. This phenomenon can be counteracted by turning down the light source intensity, but as mentioned previously, the sensor then becomes unable to respond to the low intensity light.

5.4.2 Energy Harvesting and Self Powering Performance

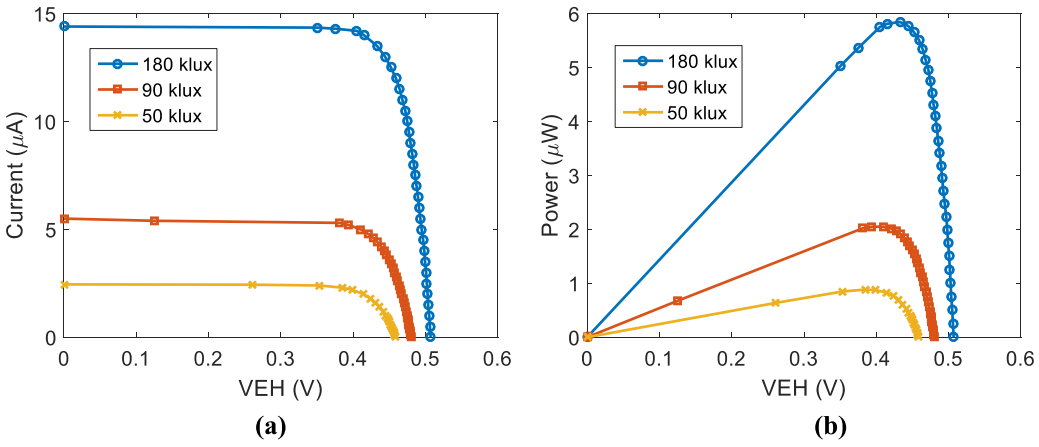


Figure 69 (a) Sensor I-V curve and (b) Generated power for the revised sensor at varying brightness levels

Figure 69 shows the sensor I-V curve and harvested power for the revised design at varying brightness levels. For the maximum brightness level, the revised sensor is able to generate 5.8µW of power. Figure 70 compares the I-V characteristics of the revised sensor array (18µm pixels) against that of the old design (9µm pixels) at 180klux illumination. Even though the open circuit voltage has increased only slightly (from 480mV to 500mV), we see a large increase in the short circuit current (14.4µA vs 6.17µA). This increase in current capacity causes the maximum power generation to increase by almost 3× to 5.8µW (from a previous value of 2.1µW). It should be noted that a 60% increase in fill factor (44% FF for 9µm pixel, 70% FF for 18µm pixel) by itself cannot be the sole reason for this large increase in power. Reduction of the well effect (Figure 56) by pushing the filler metals further away is another important catalyst for the increase in harvesting power.

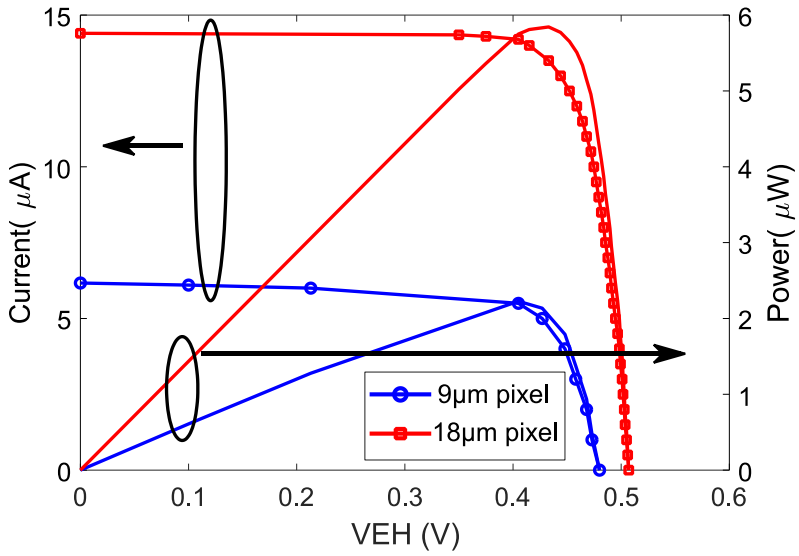


Figure 70 Comparison of harvested power between the original and revised design (180 klux brightness)

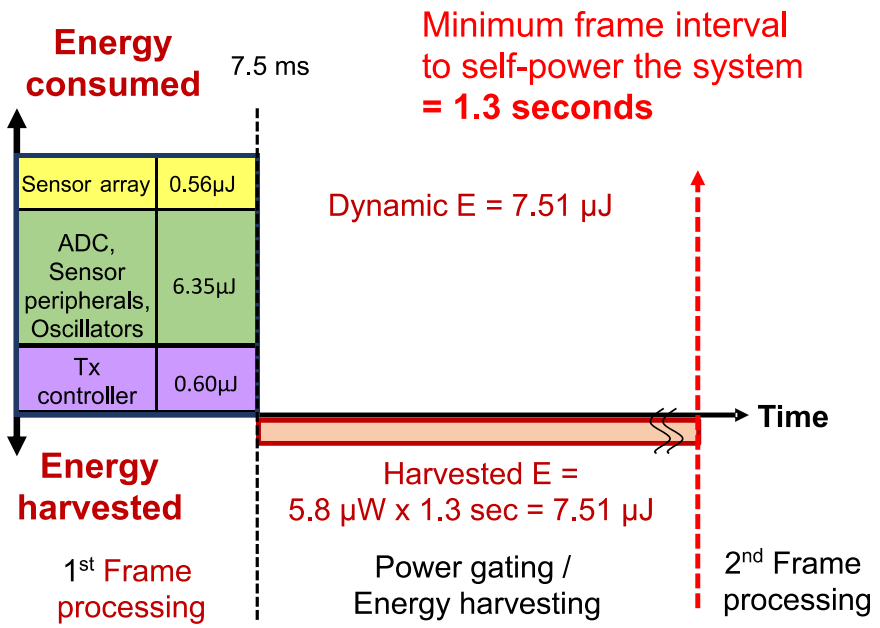


Figure 71 Breakdown of capture energy and break even point of self-sustained operation for revised chip (excluding transmitter overhead)

In order to investigate the self-powering performance of the revised system, we use one of the buck converter outputs (powered by an external source), regulating at 1.2V, to power the sensor, ADC, and transmission controller. Using the buck converter to power the chip includes the effect of power converter inefficiencies, and emulates realistic working conditions. Figure 71 shows the various components of energy for the revised testchip excluding the transmitter overhead. Due to the revised ADC architecture, and because we had to operate our system from an external clock with a maximum frequency of 2 MHz, the frame processing time (excluding transmitter) has increased to 7.5ms. However, since the transmission controller now has to deal with a much smaller data volume (due to sensor resolution reduction), its architecture can be simplified, which results in a considerable reduction in transmission controller energy. Processing a single frame consumes 7.51 µJ (compared to the previous value of 10.3 µJ) of dynamic energy, after which the system goes into harvesting mode where energy can be harvested from the

sensor. Assuming that the sensor can generate energy at the maximum value of 5.8µW, and assuming 100% power converter efficiency, our calculations show that the system will generate enough energy to process the next frame every 1.30 second, for an effective frame rate of 0.77 frames/second.

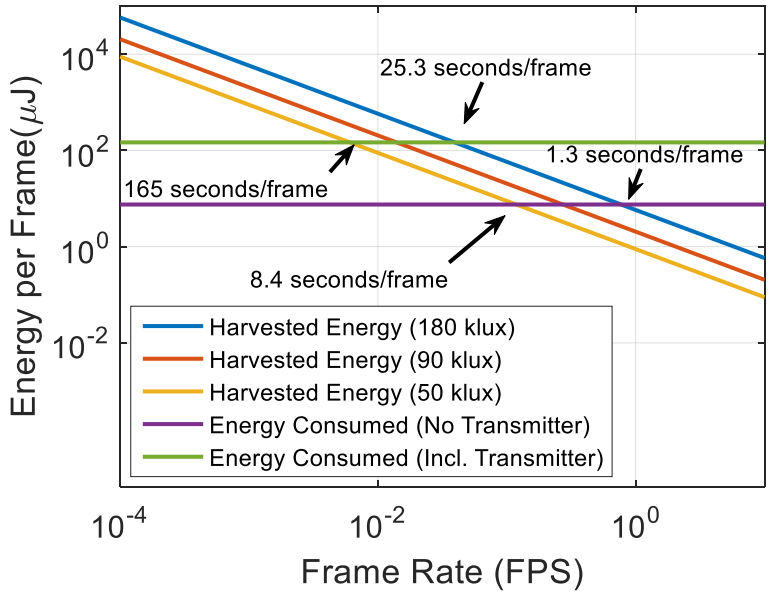


Figure 72 Harvested/consumed energy against varying frame rate and illumination for the revised chip

Figure 72 shows how the consumed and harvested energy varies with the frame rate for the revised chip. Similar to the previous analysis in Figure 51, we include the results for varying brightness and consider the effects of transmitter integration. As discussed previously, in order to achieve self-sustained operation, we must maintain a frame capture interval (the time interval between each frame capture event) longer than 1.3 seconds/frame. However, if we wish to power the transmitter with the harvested energy as well, we must consider an additional 125µJ/frame for transmitter energy. In addition, the system must stay awake for 22ms in order to interface with and send the image into the

transmitter, which increases the per frame energy consumption to 147µJ. Considering this additional transmission energy overhead, our energy neutral frame capture interval increases to 25.3 frames/second. We should note that due to reduced sensor resolution, there has been a corresponding decrease in the amount of data transmission, and hence transmission energy has also decreased compared to the previous iteration of the chip. Compared to Figure 51, this represents a significant increase in the self-sustained frame rate, primarily owing to the improved harvested power and decreased transmission power. Table 12 summarizes the self-powered frame rate data for different illumination levels.

Table 12 Key parameters of the revised testchip

Specification		**Without Transmitter**	**With Transmitter**
Maximum Power @ 180 klux (µW)		5.8	5.8
Maximum Power @ 90 klux (µW)		2.05	2.05
Maximum Power @ 50 klux (µW)		0.89	0.89
Energy Consumption per frame (µJ)	Sensor Array	0.56	1.63
	ADC, peripherals	6.35	18.6
	TX Controller	0.60	1.75
	Transmitter	N/A	125
	Total	7.51	147
Minimum Frame Capture Time (ms)		7.5	22
Self-powered frame interval @ 180 klux (s)		1.3	25.3
Self-powered frame interval @ 90 klux (s)		3.7	71.7
Self-powered frame interval @ 50 klux (s)		8.4	165

5.4.3 Autonomous Operation

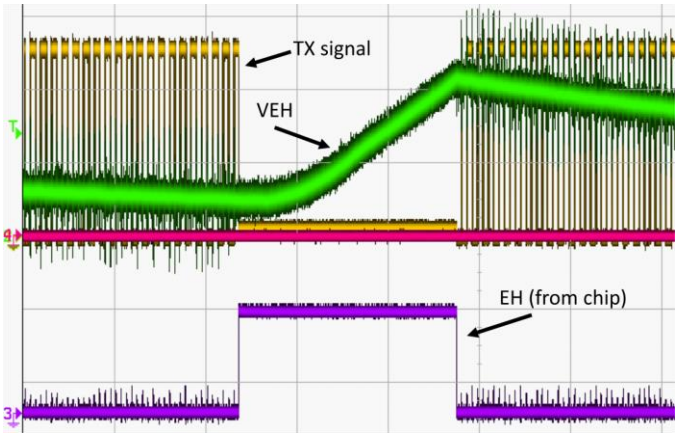


Figure 73 Oscilloscope waveform showing energy harvesting with harvesting signal generated autonomously from chip (0.2s/div, 0.2V/div for VEH)

The image sensor SOC is able to operate autonomously by automatically going into harvesting mode once it has finished processing a single frame. Figure 73 shows an example waveform of the operation. The TX signal represents the transmission (into the data acquisition unit) of each 8×8 pixel block. Once all the 48 blocks have been transmitted, the system goes into harvesting mode. At this point, the EH signal goes high and the system goes into harvesting mode, which causes the voltage at node VEH to increase. Once enough energy has been harvested from the chip, processing of the next frame can start. For this experiment, the EH signal is generated autonomously from chip but the next frame start signal is applied externally. However, the next frame start signal can also be generated autonomously through a comparator based approach by monitoring the voltage at node VEH.

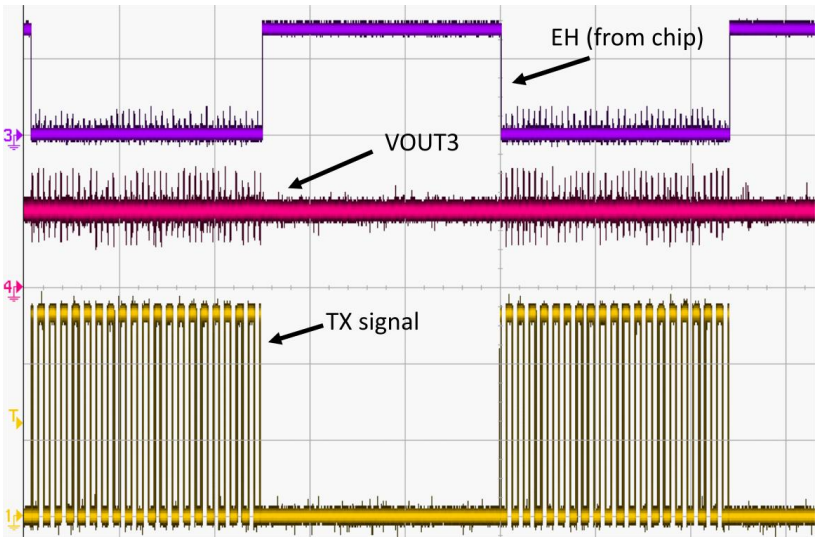


Figure 74 System operation from buck converter output rail VOUT3, regulating at 1V (0.2s/div, 1V/div for VOUT3)

Next, we try to power the sensor and logic using the on-chip power converter. For this experiment, we are going to use the third voltage rail, VOUT3, which is always on, to power the components on the chip. Figure 74 shows the buck converter operation, regulating at a voltage of 1V. The input and supply rail of the buck converter are connected to an external 3V power supply. The waveforms demonstrate proper operation even if the system is powered by the buck converter; the TX signal shows that the image has been processed and transmitted, and the harvesting signal is then generated autonomously.

5.5 Summary

This chapter explored the design of a single-chip image sensor node that can capture and process images as well as harvest energy from the on-chip pixel array. The initial version of the chip generated 2.1µW peak power at 180klux illumination, and demonstrated the feasibility of self-powered operation with an effective interval of 4.9 seconds (without transmitter) between each frame capture. However, the sensor suffered from severe noise

Table 13 Comparison of the sensor with previous work

Specification	[39]	[41]	[87]	[88]	This work - initial version	This work - revised version
Array Size	32×32	32×32	32×32	100×90	128×96	64×48
Pixel Size (μm^2)	15×15	54×48	2800×2800	5×5	9×9	18×18
Fill factor	21%	36%	N/A	46%[1], 94%[2]	44%	70%
Generated Power	35.6 nW	2 µW	0.77mW	30 µW	2.1 µW	5.8 µW

[1] Imaging photodiode, [2] Harvesting photodiode

and exhibited low photosensitivity owing to both pixel and ADC design issues. A revised version of the chip with modified sensor, ADC and power management unit was later taped out. The revised design showed significantly higher maximum power generation of 5.8µW due to increased pixel size. Table 13 summarizes the energy harvesting capabilities of the sensor and compares against previously published literature. The imaging performance was also found to be improved, with the sensor demonstrating lower noise and higher photosensitivity; however, capturing an actual pattern on the sensor proved to be challenging due to optical issues and still sub-optimal photosensitivity (which can be improved by increasing the sensor size to match the lens and switching the technology to an image sensor process). Autonomous switching between imaging and harvesting mode, as well as operation with the integrated power management unit was demonstrated. In conclusion, the presented system demonstrates a proof-of-concept for low energy, self-powered image processing nodes, and indicates a path towards implementing energy autonomous sensors for IoT applications such as remote surveillance and environmental monitoring.

CHAPTER 6. CONCLUSION

6.1 Dissertation Summary and Contributions

In this thesis, we investigated sensor integrated processing as a computing paradigm for the Internet of Things ecosystem. Transmitting large, unprocessed image data from the sensor to the host introduces latencies in the system and decreases energy efficiency by dissipating large amounts of transmission energy. Processing image (or at least part of it) on the host, however, allows the possibility of in field decision making and can potentially reduce the transmission energy to the host. This thesis investigated at the system level the implications of coupling sensors to processing engines, and evaluated the presented systems in terms of performance, energy efficiency, and accuracy under a variety of environmental conditions and constraints. In particular, we studied three types of sensor based systems. Initially, we investigated Neurosensor, a 3D-stacked image sensor coupled with a 3D integrated neural accelerator based on the Neurocube architecture. Next, we extended this architecture to enhance the parallelism of the sensor facilitated by 3D integration, and explored processing-in-memory architectures based on emerging devices for neural acceleration. Finally, we investigated post silicon results for a 2D image sensor SOC and explored power generation and delivery for these low power sensor-based systems.

In CHAPTER 3, we presented the basic architecture of Neurosensor, a 3D stacked imager with integrated neural accelerator which can perform in-field neural classification. 3D stacking helped to create a high fill factor imager by pushing the A/D conversion circuitry to the bottom layer, and increased parallelism by dividing the entire system into

16 simultaneously operating segments. Two main configurations of the system were presented – in one configuration, all synaptic weights were stored in on-chip stacked DRAM; in the other configuration, synaptic weights were stored in off-chip DDR3 memory. A coupled power, performance, thermal and noise model was developed to study the various trade-offs among performance, energy efficiency and accuracy involved with these systems. The systems were evaluated by implementing four well-known neural networks, and the implications of varying bandwidth between the sensor and the host were studied in terms of system throughput and energy consumption. The performance and energy advantages of 3D integration were made apparent through the much higher throughput and lower energy consumption for the stacked DRAM configuration. In addition, the concept of partitioned inference was also explored, where part of the DNN pipeline is implemented on chip and the partially feature extracted/classified image is then transmitted to the host in order to yield maximum energy efficiency (measured using throughput to energy ratio). In general, it was found that whenever the DNN contains cascaded fully-connected layers (e.g. AlexNet), it is often more advantageous to implement only the feature extraction layers on chip, and leave the fully connected layers to be processed on the host. Whereas if the network does not contain cascaded fully connected layers (e.g. ResNet), implementing the entire DNN on chip provides better energy efficiency. A thermal noise model was also developed to investigate the effect of temperature induced noise on neural network accuracy, and the trade-offs between system throughput and accuracy was investigated.

CHAPTER 4 extended the Neurosensor concept by exploring highly parallel high throughput imagers coupled with ReRAM based processing-in-memory architectures. The

basic imager architecture in Neurosensor was still essentially 2D in nature since it performed row-wise readout. In this chapter, we introduced a digital pixel with simplified in-pixel ADCs. Essentially, the ADCs produced spikes at a rate proportional to the photocurrent; the number of spikes produced was then counted by a digital counter, thus performing A/D conversion. Since each pixel contained its own dedicated ADC, conversion of all the pixels could take place simultaneously. This massive level of parallelism enabled the imager to achieve extremely high imaging throughputs (1.7×10^6 frames/second maximum theoretical throughput). In addition, this digital sensor also allowed performance and power to scale accordingly because of its mostly digital architecture. To take advantage of the increased imaging throughput, we investigated coupling the sensor with ISAAC, a well-known ReRAM based processing-in-memory architecture, and evaluated the system in terms of processing throughput and energy consumption. In order to gauge the impact of digital sensors as well as PIM computing, we also carried out analysis using a mix-and-match approach among analog/digital sensor and ReRAM/digital (Neurosensor) neural accelerator. Our analysis found that although classification/feature extraction is generally faster on ISAAC compared to its digital counterpart, coupling an analog sensor with ISAAC places an image capture bottleneck, thus leading to a decreased throughput. Similarly, coupling the digital sensor with the digital accelerator does not provide too much of a performance boost, since the digital accelerator is primarily limited by its relatively long processing time. In addition, we found that system memory capacity was one of the key determiners of performance for PIM based architectures. The performance advantages of PIM can only be utilized if all the synaptic weights can be stored on chip; any time the weights have to be fetched from off-chip

DRAM, we are not doing processing-in-computing any more, and off-chip memory access often entails a drastic drop in energy efficiency.

CHAPTER 5 presented the design of a 2D image sensor SOC. In addition to the standard components in a wireless image sensor node (sensor, ADC, memory, processor and transmission controller), the system also contained an integrated power management unit whose function was to provide power to the system blocks in imaging mode and harvest energy from the image sensor array in harvesting mode. With the overall goal being energy neutrality, where the sensor is able to operate based solely on energy harvested from the chip, the sensor was found to generate a peak power of 2.1μW, which equates to an image capture interval of 4.90 seconds for energy neutral operation (assuming 100% power converter efficiency, no transmission overhead, and maximum power point harvesting). However, the imaging performance was found to be quite noisy and insensitive to light, which made it difficult to gain any meaningful image information from the sensor. Further analysis was carried out to identify the main sources of the shortcomings, and a revised version of the testchip was taped out which implemented modifications (bigger pixels, high threshold harvesting transistor, central ramp generator for ADC) to remedy the above problems. The revised testchip showed significantly higher power generation – up to 5.8 μW, which caused the self-powered frame rate to increase considerably to 1.3 seconds (considering no transmission overhead, maximum power point harvesting), and also exhibited better photosensitivity in imaging mode which allowed the creation of simple patterns using the image sensor. The testchip was also demonstrated to exhibit autonomous operation, with the sensor going to the harvesting mode automatically once a captured frame had finished processing and transmitting.

6.2 Future Work

There are several open avenues available to continue this research further. As a first step, we can extend the work on the PIM architecture in CHAPTER 4. We based our results on a relatively simplistic model, as opposed to a detailed cycle level simulator approach similar to that carried out with Neurosensor. Developing the model further to include more detailed latency and energy information (e.g. latency and energy associated with a crossbar operation, eDRAM access, ADC/DAC power, and so on) would serve to provide a more accurate estimate of the performance and energy of PIM accelerators.

One of the main overheads in ReRAM based neural accelerators comes from ADC/DAC power. In addition, since the ReRAMs have finite resistance, the wordlines require opamp buffers to sink large amounts of current. FeFET based crossbar arrays have the potential to be used as memory due to multiple threshold voltage states, do not require ADC/DACs, present high (almost infinite) resistance to the wordline driver (thus not needing a buffer), and have lower programming energy than ReRAM. Thus replacing the ReRAM based PIM computing layer with an FeFET based one has the potential to further improve energy efficiency and performance.

The self-powering analysis in CHAPTER 5 presents an interesting concept towards the operation of low power systems where they can be kept functional solely on harvested energy (provided the frame rate is kept low). This concept can also be extended to 3D-stacked sensors integrated with neural accelerators. Since the 3D sensors generally have higher fill factor than their 2D counterparts, these systems have even greater potential to generate more energy and remain operational with harvested energy. However, since the

power consumption of these systems are also typically higher compared to the image sensor node, proper analysis needs to be carried out regarding the feasibility of operating 3D stacked neural accelerators with harvested power.

REFERENCES

[1] B. Black, M. Annavaram, N. Brekelbaum, *et al.*, "Die Stacking (3D) Microarchitecture," in *2006 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'06)*, 2006, pp. 469-479.

[2] K. Banerjee, S. J. Souri, P. Kapur, *et al.*, "3-D ICs: a novel chip design for improving deep-submicrometer interconnect performance and systems-on-chip integration," *Proceedings of the IEEE*, vol. 89, pp. 602-633, 2001.

[3] W. R. Davis, J. Wilson, S. Mick, *et al.*, "Demystifying 3D ICs: the pros and cons of going vertical," *IEEE Design & Test of Computers*, vol. 22, pp. 498-510, 2005.

[4] C. h. Yu, "The 3rd dimension-More Life for Moore's Law," in *2006 International Microsystems, Package, Assembly Conference Taiwan*, 2006, pp. 1-6.

[5] Z. Fu and E. Culurciello, "A 3D Integrated Feature-Extracting Image Sensor," in *2007 IEEE International Symposium on Circuits and Systems*, 2007, pp. 3964-3967.

[6] S. F. Yeh, C. C. Hsieh, and K. Y. Yeh, "A 3 Megapixel 100 Fps 2.8 μm Pixel Pitch CMOS Image Sensor Layer With Built-in Self-Test for 3D Integrated Imagers," *IEEE Journal of Solid-State Circuits*, vol. 48, pp. 839-849, 2013.

[7] V. Suntharalingam, R. Berger, J. A. Burns, *et al.*, "Megapixel CMOS image sensor fabricated in three-dimensional integrated circuit technology," in *ISSCC. 2005 IEEE International Digest of Technical Papers. Solid-State Circuits Conference, 2005.*, 2005, pp. 356-357 Vol. 1.

[8] M. J. Wolf, P. Ramm, A. Klumpp, *et al.*, "Technologies for 3D wafer level heterogeneous integration," in *2008 Symposium on Design, Test, Integration and Packaging of MEMS/MOEMS*, 2008, pp. 123-126.

[9] C. Ting-Yen, S. J. Souri, C. Chi On, *et al.*, "Thermal analysis of heterogeneous 3D ICs with various integration scenarios," in *International Electron Devices Meeting. Technical Digest (Cat. No.01CH37224)*, 2001, pp. 31.2.1-31.2.4.

[10] D. Choudhury, "3D integration technologies for emerging microsystems," in *2010 IEEE MTT-S International Microwave Symposium*, 2010, pp. 1-4.

[11] K. Kiyoyama, Y. Sato, H. Hashimoto, *et al.*, "A block-parallel ADC with digital noise cancelling for 3-D stacked CMOS image sensor," in *2013 IEEE International 3D Systems Integration Conference (3DIC)*, 2013, pp. 1-4.

[12] Y. Ohara, K. W. Lee, K. Kiyoyama, *et al.*, "Chip-based hetero-integration technology for high-performance 3D stacked image sensor," in *2012 2nd IEEE CPMT Symposium Japan*, 2012, pp. 1-4.

[13] K. W. Lee, Y. Ohara, K. Kiyoyama, *et al.*, "Die-Level 3-D Integration Technology for Rapid Prototyping of High-Performance Multifunctionality Hetero-Integrated Systems," *IEEE Transactions on Electron Devices*, vol. 60, pp. 3842-3848, 2013.

[14] X. Zhang, S. Chen, and E. Culurciello, "A second generation 3D integrated feature-extracting image sensor," in *2011 IEEE SENSORS Proceedings*, 2011, pp. 1933-1936.

[15] S. Sukegawa, T. Umebayashi, T. Nakajima, *et al.*, "A 1/4-inch 8Mpixel back-illuminated stacked CMOS image sensor," in *2013 IEEE International Solid-State Circuits Conference Digest of Technical Papers*, 2013, pp. 484-485.

[16] D. Lie, K. Chae, and S. Mukhopadhyay, "Analysis of the Performance, Power, and Noise Characteristics of a CMOS Image Sensor With 3-D Integrated Image Compression Unit," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 4, pp. 198-208, 2014.

[17] D. Lie, A. R. Trivedi, and S. Mukhopadhyay, "Impact of Heterogeneous Technology Integration on the Power, Performance, and Quality of a 3D Image Sensor," *IEEE Transactions on Multi-Scale Computing Systems*, vol. 2, pp. 61-67, 2016.

[18] T. Haruta, T. Nakajima, J. Hashizume, *et al.*, "A 1/2.3inch 20Mpixel 3-layer stacked CMOS Image Sensor with DRAM," in *2017 IEEE International Solid-State Circuits Conference Digest of Technical Papers*, 2017, pp. 76-77.

[19] K. I. Schultz, M. W. Kelly, J. J. Baker, *et al.*, "Digital-Pixel Focal Plane Array Technology," *Lincoln Laboratory Journal*, vol. 20, pp. 36-51, 2014.

[20] M. Goto, K. Hagiwara, Y. Honda, *et al.*, "128x96 Pixel-parallel three-dimensional integrated CMOS image sensors with 16-bit A/D converters: By direct bonding with embedded Au electrodes," in *2015 IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference (S3S)*, 2015, pp. 1-2.

[21] M. Sakakibara, K. Ogawa, S. Sakai, *et al.*, "A Back-Illuminated Global-Shutter CMOS Image Sensor with Pixel-Parallel 14b Subthreshold ADC," in *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2018 IEEE International*, 2018, pp. 80-81.

[22] S. Han, J. Pool, J. Tran, *et al.*, "Learning both weights and connections for efficient neural networks," *28th International Conference on Neural Information Processing Systems - Volume 1*, pp. 1135-1143, 2015.

[23] T. Chen, Z. Du, N. Sun, *et al.*, "DianNao: a small-footprint high-throughput accelerator for ubiquitous machine-learning," *SIGPLAN Not.*, vol. 49, pp. 269-284, 2014.

[24] Y. Chen, T. Luo, S. Liu, *et al.*, "DaDianNao: A Machine-Learning Supercomputer," in *2014 47th Annual IEEE/ACM International Symposium on Microarchitecture*, 2014, pp. 609-622.

[25] Z. Du, R. Fasthuber, T. Chen, *et al.*, "ShiDianNao: Shifting vision processing closer to the sensor," in *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, 2015, pp. 92-104.

[26] B. Belhadj, A. Valentian, P. Vivet, *et al.*, "The improbable but highly appropriate marriage of 3D stacking and neuromorphic accelerators," in *2014 International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES)*, 2014, pp. 1-9.

[27] D. Kim, J. Kung, S. Chai, *et al.*, "Neurocube: A Programmable Digital Neuromorphic Architecture with High-Density 3D Memory," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, 2016, pp. 380-392.

[28] J. Jeddeloh and B. Keeth, "Hybrid memory cube new DRAM architecture increases density and performance," in *2012 Symposium on VLSI Technology (VLSIT)*, 2012, pp. 87-88.

[29] S. Gould, R. Fulton, and D. Koller, "Decomposing a scene into geometric and semantically consistent regions," in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 1-8.

[30] M. Gao, J. Pu, X. Yang, *et al.*, "TETRIS: Scalable and Efficient Neural Network Acceleration with 3D Memory," in *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems*, Xi'an, China, 2017, pp. 751-764.

[31] E. Azarkhish, D. Rossi, I. Loi, *et al.*, "Neurostream: Scalable and Energy Efficient Deep Learning with Smart Memory Cubes," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, pp. 420-434, 2018.

[32] H. Akinaga and H. Shima, "Resistive Random Access Memory (ReRAM) Based on Metal Oxides," *Proceedings of the IEEE*, vol. 98, pp. 2237-2251, 2010.

[33] A. Shafiee, A. Nag, N. Muralimanohar, *et al.*, "ISAAC: a convolutional neural network accelerator with in-situ analog arithmetic in crossbars," *SIGARCH Comput. Archit. News*, vol. 44, pp. 14-26, 2016.

[34] P. Chi, S. Li, C. Xu, *et al.*, "PRIME: a novel processing-in-memory architecture for neural network computation in ReRAM-based main memory," in *Proceedings of*

the 43rd International Symposium on Computer Architecture, Seoul, Republic of Korea, 2016, pp. 27-39.

[35] L. Song, X. Qian, H. Li, *et al.*, "PipeLayer: A Pipelined ReRAM-Based Accelerator for Deep Learning," in *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2017, pp. 541-552.

[36] G. Kim, Y. Lee, F. Zhiyoong, *et al.*, "A millimeter-scale wireless imaging system with continuous motion detection and energy harvesting," in *2014 Symposium on VLSI Circuits Digest of Technical Papers*, 2014, pp. 1-2.

[37] A. Fish, S. Hamami, and O. Yadid-Pecht, "CMOS Image Sensors With Self-Powered Generation Capability," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 53, pp. 1210-1214, 2006.

[38] C. Shi, M. K. Law, and A. Bermak, "A Novel Asynchronous Pixel for an Energy Harvesting CMOS Image Sensor," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, pp. 118-129, 2011.

[39] M. K. Law, A. Bermak, and C. Shi, "A Low-Power Energy-Harvesting Logarithmic CMOS Image Sensor With Reconfigurable Resolution Using Two-Level Quantization Scheme," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 58, pp. 80-84, 2011.

[40] S. U. Ay, "A CMOS Energy Harvesting and Imaging (EHI) Active Pixel Sensor (APS) Imager for Retinal Prosthesis," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 5, pp. 535-545, 2011.

[41] H. T. Wang and W. D. Leon-Salas, "An Image Sensor With Joint Sensing and Energy Harvesting Functions," *IEEE Sensors Journal*, vol. 15, pp. 902-916, 2015.

[42] H. S. Wong, "Technology and device scaling considerations for CMOS imagers," *IEEE Transactions on Electron Devices*, vol. 43, pp. 2131-2142, 1996.

[43] S. Kavadias, B. Dierickx, D. Scheffer, *et al.*, "A logarithmic response CMOS image sensor with on-chip calibration," *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 1146-1152, 2000.

[44] D. Joseph and S. Collins, "Transient response and fixed pattern noise in logarithmic CMOS image sensors," *IEEE Sensors Journal*, vol. 7, pp. 1191-1199, 2007.

[45] T. Sugiki, S. Ohsawa, H. Miura, *et al.*, "A 60 mW 10 b CMOS image sensor with column-to-column FPN reduction," in *2000 IEEE International Solid-State Circuits Conference. Digest of Technical Papers (Cat. No.00CH37056)*, 2000, pp. 108-109.

[46] J. H. Ko, M. F. Amir, K. Z. Ahmed, *et al.*, "A Single-Chip Image Sensor Node with Energy Harvesting from a CMOS Pixel Array," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, pp. 2295-2307, 2017.

[47] *Nangate FreePDK15 Open Cell Library*. Available: http://www.nangate.com/?page_id=2328

[48] J. Hu and R. Marculescu, "DyAD: smart routing for networks-on-chip," in *41st annual Design Automation Conference*, San Diego, CA, USA, 2004, pp. 260-263.

[49] Y. Lecun, L. Bottou, Y. Bengio, *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278-2324, 1998.

[50] S. Lawrence, C. L. Giles, T. Ah Chung, *et al.*, "Face recognition: a convolutional neural-network approach," *IEEE Transactions on Neural Networks*, vol. 8, pp. 98-113, 1997.

[51] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems*, Lake Tahoe, Nevada, 2012, pp. 1097-1105.

[52] K. Simonyan and A. Zisserman. (2014, October 3, 2017). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv e-prints 1409*. Available: https://arxiv.org/abs/1409.1556

[53] C. Szegedy, W. Liu, Y. Jia, *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1-9.

[54] K. He, X. Zhang, S. Ren, *et al.*, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770-778.

[55] M. F. Amir, J. H. Ko, T. Na, *et al.*, "3-D Stacked Image Sensor with Deep Neural Network Computation," *IEEE Sensors Journal*, vol. 18, pp. 4187-4199, 2018.

[56] E. Karl, Z. Guo, J. W. Conary, *et al.*, "A 0.6V 1.5GHz 84Mb SRAM design in 14nm FinFET CMOS technology," in *2015 IEEE International Solid-State Circuits Conference Digest of Technical Papers*, 2015, pp. 310-311.

[57] S. Das, A. Chandrakasan, and R. Reif, "Timing, energy, and thermal performance of three-dimensional integrated circuits," in *Proceedings of the 14th ACM Great Lakes symposium on VLSI*, Boston, MA, USA, 2004, pp. 338-343.

[58] K. Puttaswamy and G. H. Loh, "Thermal analysis of a 3D die-stacked high-performance microprocessor," in *Proceedings of the 16th ACM Great Lakes symposium on VLSI*, Philadelphia, PA, USA, 2006, pp. 19-24.

[59] International Technology Roadmap for Semiconductors 2011. Available: http://www.itrs.net

[60] D. Joseph and S. Collins, "Temperature Dependence of Fixed Pattern Noise in Logarithmic CMOS Image Sensors," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, pp. 2503-2511, 2009.

[61] K. T. Malladi, B. C. Lee, F. A. Nothaft, *et al.*, "Towards energy-proportional datacenter memory with mobile DRAM," in *Proceedings of the 39th Annual International Symposium on Computer Architecture*, Portland, Oregon, 2012, pp. 37-48.

[62] Ralink. MT7620 Datasheet. Available: http://www.datasheet.fr/PDF/788206/MT7620-pdf.html

[63] NordicSemiconductor. nRF24L01+ Single Chip 2.4 GHz Transceiver Preliminary Product Specification. Available: https://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01P

[64] J. H. Ko, "Resource-aware and Robust Image Processing for Intelligent Sensor Systems," PhD Thesis, Georgia Institute of Technology, 2018.

[65] Y. Jia, E. Shelhamer, J. Donahue, *et al.*, "Caffe: Convolutional Architecture for Fast Feature Embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, Orlando, Florida, USA, 2014, pp. 675-678.

[66] M. F. Amir, D. Kim, J. Kung, *et al.*, "NeuroSensor: A 3D image sensor with integrated neural accelerator," in *2016 IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference (S3S)*, 2016, pp. 1-2.

[67] M. Z. Kuo, O. Takahashi, P. L. Yang, *et al.*, "A HKMG 28nm 1GHz fully-pipelined tile-able 1MB embedded SRAM IP with 1.39mm2 per MB," in *Proceedings of the IEEE 2013 Custom Integrated Circuits Conference*, 2013, pp. 1-4.

[68] B. Akin, F. Franchetti, and J. C. Hoe, "Data reorganization in memory using 3D-stacked DRAM," in *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, 2015, pp. 131-143.

[69] J. Ahn, S. Hong, S. Yoo, *et al.*, "A scalable processing-in-memory accelerator for parallel graph processing," in *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, 2015, pp. 105-117.

[70] D. Zhang, N. Jayasena, A. Lyashevsky, *et al.*, "TOP-PIM: throughput-oriented programmable processing in memory," in *Proceedings of the 23rd international symposium on High-performance parallel and distributed computing*, Vancouver, BC, Canada, 2014, pp. 85-98.

[71] Y. Long, T. Na, and S. Mukhopadhyay, "ReRAM based Processing-in-memory Architecture for Recurrent Neural Network Acceleration (Accepted for Publication)," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*.

[72] H. S. P. Wong, H. Y. Lee, S. Yu, *et al.*, "Metal–Oxide RRAM," *Proceedings of the IEEE*, vol. 100, pp. 1951-1970, 2012.

[73] B. Gao, Y. Bi, H.-Y. Chen, *et al.*, "Ultra-Low-Energy Three-Dimensional Oxide-Based Electronic Synapses for Implementation of Robust High-Accuracy Neuromorphic Computation Systems," *ACS Nano*, vol. 8, pp. 6998-7004, 2014/07/22 2014.

[74] Y. V. Pershin and M. D. Ventra, "Experimental demonstration of associative memory with memristive neural networks," *Neural Netw.*, vol. 23, pp. 881-886, 2010.

[75] K.-H. Kim, S. Gaba, D. Wheeler, *et al.*, "A Functional Hybrid Memristor Crossbar-Array/CMOS System for Data Storage and Neuromorphic Applications," *Nano Letters*, vol. 12, pp. 389-395, 2012/01/11 2012.

[76] K. Z. Ahmed, M. F. Amir, J. H. Ko, *et al.*, "Reconfigurable 96x128 active pixel sensor with 2.1μW/mm2 power generation and regulated multi-domain power delivery for self-powered imaging," in *ESSCIRC Conference 2016: 42nd European Solid-State Circuits Conference*, 2016, pp. 507-510.

[77] K. Z. Ahmed, "Efficient Power Management Circuits for Energy Harvesting Applications," PhD Thesis, Georgia Institute of Technology, 2016.

[78] T. Esram and P. L. Chapman, "Comparison of Photovoltaic Array Maximum Power Point Tracking Techniques," *IEEE Transactions on Energy Conversion*, vol. 22, pp. 439-449, 2007.

[79] B. Subudhi and R. Pradhan, "A Comparative Study on Maximum Power Point Tracking Techniques for Photovoltaic Power Systems," *IEEE Transactions on Sustainable Energy*, vol. 4, pp. 89-98, 2013.

[80] K. Z. Ahmed and S. Mukhopadhyay, "A 110nA synchronous boost regulator with autonomous bias gating for energy harvesting," in *Proceedings of the IEEE 2013 Custom Integrated Circuits Conference*, 2013, pp. 1-4.

[81] K. Z. Ahmed and S. Mukhopadhyay, "A 190 nA Bias Current 10 mV Input Multistage Boost Regulator With Intermediate-Node Control to Supply RF Blocks in Self-Powered Wireless Sensors," *IEEE Transactions on Power Electronics*, vol. 31, pp. 1322-1333, 2016.

[82] NordicSemiconductor. nRF52840 Product Specification v1.0. Available: http://infocenter.nordicsemi.com/pdf/nRF52840_PS_v1.0.pdf

[83] IBM, "IBM CMRF8SF Design Manual."

[84] P. M. Nadeau, A. Paidimarri, and A. P. Chandrakasan, "Ultra Low-Energy Relaxation Oscillator With 230 fJ/cycle Efficiency," *IEEE Journal of Solid-State Circuits*, vol. 51, pp. 789-799, 2016.

[85] Sunex DSL218 Specification Sheet. Available: http://www.optics-online.com/OOL/DSL/DSL218.PDF

[86] *Arducam 2MP V2 Mini Camera Shield w/ ESP8266 Nano Module*. Available: https://www.robotshop.com/en/arducam-2mp-v2-mini-camera-shield-esp8266-nano-module.html

[87] S. K. Nayar, D. C. Sims, and M. Fridberg, "Towards Self-Powered Cameras," in *2015 IEEE International Conference on Computational Photography (ICCP)*, 2015, pp. 1-10.

[88] S. Y. Park, K. Lee, H. Song, *et al.*, "Simultaneous Imaging and Energy Harvesting in CMOS Image Sensor Pixels," *IEEE Electron Device Letters*, vol. 39, pp. 532-535, 2018.